Titre: Title:	Unifying Geometry and Mesh Adaptive Refinement Using Loop Subdivision
Auteur: Author:	Man Wang
Date:	2013
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Wang, M. (2013). Unifying Geometry and Mesh Adaptive Refinement Using Loop Subdivision [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/1184/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/1184/
Directeurs de recherche: Advisors:	François Guibault
Programme: Program:	Génie informatique

UNIVERSITÉ DE MONTRÉAL

UNIFYING GEOMETRY AND MESH ADAPTIVE REFINEMENT USING LOOP SUBDIVISION

MAN WANG

DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION

DU DIPLÔME DE PHILOSOPHIAE DOCTOR

(GÉNIE INFORMATIQUE)

JUILLET 2013

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

UNIFYING GEOMETRY AND MESH ADAPTIVE REFINEMENT USING LOOP SUBDIVISION

présentée par : WANG Man

en vue de l'obtention du diplôme de : Philosophiae Doctor

a été dûment acceptée par le jury d'examen constitué de :

Mme CHERIET Farida, Ph.D., présidente

M. GUIBAULT François, Ph.D., membre et directeur de recherche

M. CAMARERO Ricardo, Ph.D., membre

M. KHACHAN Mohammed, Ph.D., membre

DEDICATION

To my family,
especially to my husband David,
for his unequaled love and support!

ACKNOWLEGEMENTS

Writing this thesis is a big challenge for me. First of all, I do appreciate that my research director François Guibault and the research associate Marie-Gabrielle Vallet are here for me. Even though there are always a lot of corrections and feedbacks from them, I understand their rigorous academic spirits are very important for scientific researches and writings. Eventually, my research and knowledge have been improved by their persistent efforts. I also do appreciate their recommendations of refinement criteria and mesh quality evaluations. The achieved results and contributions of this research are largely due to their guidance.

Here, I would like to thank again my research director François Guibault particularly for offering me this opportunity to pursue my Ph.D. dream and also for his support and encouragement during the whole period at École Polychnique. This period means a lot in my life, especially starting my family at the same time.

I would like to thank that all the members of the jury for giving me the opportunity to present our research approach and for taking times to read my thesis.

I also would like to thank Mr. Julien Dompierre particularly for providing the documents and source codes related to the ETA evaluations, and for correcting the part ‘condensé en français’.

I also want to take this opportunity to express my appreciations to my friends at École Polytechnique, particularly Mélissa Côté and Éric Joubarne. Many thanks must go to Mélissa, for her suggestions and corrections in the part ‘condensé en français’ and Chapter 1 and Chapter 2, for sharing the Ph.D. study experiences with me, also for her encouragement especially at the most difficult stages of my Ph.D. study.

Here, I would like to particularly thank my sister-in-laws Isabelle Chabot for her tremendous support and encouragement in many ways.

Pursuing Ph.D. studies is a very long journey for me, and my family. I also would like to take this opportunity to thank my husband David Chabot, for his understanding, support and love.

ABSTRACT

In this thesis, we present a new refinement approach on three types of meshes: curves, triangular surfaces and 3D tetrahedral meshes. This approach utilizes subdivision-based representations to create, modify, analyze and visualize geometric models with arbitrary topology for numerical simulation applications. The subdivision-based representations are generated by utilizing Loop subdivisions. After studying the disadvantage of lack of flexibility in controlling LODs (Level Of Details) and accuracy in representing geometric models by using the non-uniform approximating subdivision iterations to approach simulated models, we introduce adaptive subdivisions in our refinement work.

We develop a single-level refinement method to support adaptive subdivisions on the three types of meshes. This single-level method eliminates the hierarchy storage and the stitching issues encountered during the generation of multi-resolution subdivision meshes, especially 3D tetrahedral meshes. The implementation of adaptive tetrahedral mesh subdivisions brings up two innovations: the configuration of tetrahedron split patterns and the improvement in subdivision surface parameterizations. The natural combination of these two innovations fulfills generating multi-resolution subdivision tetrahedral meshes, whose boundary surfaces lie exactly on their subdivision limits.

Our research work includes five parts. Firstly, we develop the Loop-based solid subdivision scheme, which permits integrating edge-based topological splits with geometrical smoothing on boundary surfaces. Secondly, we merge subdivision techniques with adaptive refinements with, which permits whole meshes to be adaptively subdivided and boundary meshes to be projected to their subdivision limits. Thirdly, we study and compare the existing subdivision surface parameterization techniques, which eventually permits obtaining the limit subdivision of any arbitrary position on Loop subdivision surfaces. Fourthly, we complete vertex and edge crease creation rules of the Loop-based solid subdivision scheme, which permits preserving sharp features on boundary surfaces of 3D tetrahedral meshes. Finally, we use a mesh quality evaluator to validate our results and we evaluate system performance in the context of solid modeling.

RÉSUMÉ

Cette thèse présente une nouvelle approche pour le raffinement de trois types de maillages : courbes, surfaces triangulaires et maillages tétraédriques tridimensionnels. Cette approche utilise des représentations par subdivisions afin de définir, modifier, analyser et visualiser des modèles géométriques de topologie arbitraire pour les applications de simulation numérique. Les représentations par subdivisions sont générées à l'aide des subdivisions de Loop. Après avoir étudié les inconvénients du manque de flexibilité dans le contrôle des niveaux de détails et du manque de précision dans les représentations de modèles géométriques utilisant les subdivisions itératives, approximatives et non-uniformes pour se rapprocher des modèles simulés, nous introduisons une nouvelle méthode de subdivision adaptative pour le raffinement de maillages.

Cette méthode de raffinement à un seul niveau a été développée afin de supporter les subdivisions adaptatives pour les trois types de maillages. Cette méthode évite le stockage par hiérarchie et les problèmes d'assemblage rencontrés durant la génération des maillages multi-résolutions par subdivisions, surtout pour les maillages tétraédriques. La mise en œuvre de subdivisions pour les maillages adaptatifs tétraédriques amène deux innovations : la configuration de forme de fractionnement des tétraèdres et l'amélioration de la paramétrisation des surfaces de subdivision. La combinaison naturelle de ces deux innovations permet la génération par subdivision de maillages multi-résolutions tétraédriques dont les surfaces frontières sont exactement sur les limites de subdivision.

Notre recherche contient cinq parties. Premièrement, nous développons un schéma de Loop pour la subdivision des solides, lequel permet d'intégrer le fractionnement topologique des arêtes avec le lissage géométrique des surfaces frontières. Deuxièmement, nous fusionnons les raffinements adaptatifs avec les techniques de subdivision, ce qui permet la subdivision adaptive complète du maillage tout en ayant les surfaces frontières projetées sur les limites de subdivision. Troisièmement, nous étudions et comparons des techniques existantes de paramétrisation des surfaces de subdivision, ce qui permet d'obtenir directement la limite de subdivision de toutes positions arbitraires sur les surfaces de subdivision de Loop. Quatrièmement, nous construisons les règles de création des sommets fixes et des arêtes vives du schéma de subdivision de Loop pour les modèles solides, ce qui permet de préserver les caractéristiques anguleuses des surfaces frontières des maillages tétraédriques. Finalement, nous utilisons un critère de qualité des

maillages pour valider nos résultats et nous présentons la performance des calculs en ce qui a trait à la modélisation des solides.

CONDENSÉ EN FRANÇAIS

1. Introduction

La simulation de phénomènes physiques complexes sur des objets réels est très en demande dans de nombreux champs d'application de l'ingénierie, comme l'animation basée sur la physique, la simulation en neurochirurgie, la conception et le design industriel. Habituellement, un modèle d'analyse d'ingénierie comprend deux aspects : la représentation géométrique du domaine considéré et les modèles physiques sous forme d'équations aux dérivées partielles (PDEs). La description géométrique précise des objets et la modélisation exacte de phénomènes physiques constituent donc deux champs complémentaires de recherche qui une fois réunis permettent des simulations numériques.

Le processus de simulation numérique est tributaire de différentes disciplines : la modélisation physique, la modélisation géométrique, la génération de maillages, l'analyse numérique et la visualisation scientifique. La plupart du temps, ces cinq disciplines connexes dans l'utilisation de simulations numériques réelles sont traitées comme des champs de recherche indépendants ayant chacun leur propres méthodes et représentations. Lorsque considérés indépendamment les uns des autres, des efforts additionnels sont nécessaires afin de réconcilier les incompatibilités inhérentes à chaque champ et peuvent introduire des inexactitudes durant les transitions.

De manière plus spécifique, nous pourrions dire que le processus de simulation est basé sur une capacité adéquate pour la définition, la représentation et la modification géométrique. Durant le processus de recherche d'une solution numérique utilisant l'adaptation de maillages, une question est soulevée : d'une part, est-ce que le nouveau maillage adapté est conforme à la géométrie initiale définie lors de la modélisation et d'autre part, comment peut-on s'assurer de la conformité entre les deux.

Dans le but d'améliorer la précision et l'efficacité des simulations numériques, nous nous proposons de chercher une représentation géométrique unifiée qui ne serve pas seulement de base commune durant tout le processus de simulation mais qui intègre également géométrie et maillage comme support au raffinement des maillages et à la modification géométrique.

À l'heure actuelle, deux types de représentations géométriques sont normalement utilisés: Une représentation analytique sous forme d'un modèle paramétrique des frontières (B-rep) et une représentation discrète sous forme d'un maillage polygonal des frontières.

Le principal avantage de l'utilisation de la représentation paramétrique de frontières (B-rep) est la possibilité de manipuler ou de reconstruire les représentations paramétriques par interpolation ou approximation. De plus, l'évaluation géométrique comme la courbure ou les interrogations des dérivées est possible avec les modèles (B-rep). Il est toutefois difficile de maintenir la continuité entre les différentes modifications ou frontières avec ce modèle particulièrement pour ceux ayant une topologie arbitraire ou une géométrie complexe. Dans le contexte de l'adaptation près des frontières du domaine, les problèmes liés à la préservation des arêtes vives, à la définition continue des frontières du domaine et à la rapidité de la reconstruction des régions locales affectées par les adaptations sont très complexes.

Les maillages de polygones en tant que représentations géométriques sont très largement utilisés pour représenter les modèles géométriques de topologie arbitraire en simulations numériques. La principale difficulté d'utilisation des maillages de polygones réside tout d'abord dans la manipulation d'importantes quantités de données en temps réel et aussi dans le maintien et l'évaluation de la continuité de la surface.

Afin d'atténuer les difficultés inhérentes à l'utilisation traditionnelle des représentations paramétriques des frontières et à leur discrétisation en maillage de polygones, nous proposons d'utiliser une représentation multi-résolution de subdivision en tant que représentation géométrique unifiée. Celle-ci permet de représenter la géométrie sous-jacente et supporte les raffinements des maillages ainsi que la modification géométrique pour l'analyse d'ingénierie. Nous croyons que l'intégration de représentations multi-résolution de subdivision va permettre d'unifier les analyses numériques, la modélisation géométrique, les opérations de raffinement et de visualisation ce qui permettra d'éviter l'accumulation d'erreurs et les efforts additionnels requis afin de combler les incompatibilités.

Les représentations basées sur la subdivision combinent les avantages des maillages de polygones et des représentations paramétriques (B-rep). Cet aspect souligne la possibilité unique de représenter les modèles géométriques lisses ayant une topologie arbitraire, supportant les raffinements tout en préservant les détails géométriques, maintenant la continuité de la surface

durant les modifications de forme, fournissant des valeurs géométriques pour l'analyse d'ingénierie comme les valeurs des dérivées et des courbures selon des paramètres arbitraires et préservant en plus les particularités géométriques telles que les arêtes vives définies par les usagers.

Les défis d'utilisation des représentations utilisant la subdivision de Loop peuvent être énumérés comme étant 1) le calcul des limites est particulièrement difficile pour les maillages lorsque qu'un triangle contient plus d'un sommet extraordinaire ou un sommet fixe ou une arête vive entre deux sommets, 2) les itérations uniformes approximatives selon la méthode traditionnelle de raffinement par subdivision sont coûteuses en ressources informatiques tout en manquant de flexibilité en ce qui a trait au contrôle et à la localité des détails requis, 3) les subdivisions volumétriques n'ont pas encore été bien définies et cette technique n'a pas encore eu un impact clair sur les applications d'ingénierie, 4) les problèmes les plus communs issus de l'utilisation des surfaces de subdivision pour l'analyse d'ingénierie (Gonsor and Neamtu 2001; Cirak, Scott et al. 2002; Ma 2005) tel que la paramétrisation de surface, la représentation des courbures, la continuité des surfaces et le comportement près des sommets extraordinaires.

Le but principal de notre recherche est de développer une nouvelle méthode de raffinement en unifiant la géométrie et l'adaptation de maillage aux fins d'analyse d'ingénierie. Cette méthode de raffinement est basée sur les subdivisions de Loop. Les représentations multi-résolution de subdivisions générées par notre nouvelle méthode de raffinement respecteront non seulement les besoins géométriques mais permettront en plus d'améliorer la précision de la représentation. L'objectif général repose sur les deux hypothèses suivantes:

H1: La question à savoir comment obtenir le maillage de contrôle initial retraçant une surface limite de subdivision est pré-résolue.

H2: Les critères utilisés durant le raffinement adaptatif peuvent être définies par les utilisateurs selon des besoins concrets d'analyse d'ingénierie dans les applications réels.

En utilisant les deux hypothèses ci-dessus, les sous-objectifs concrets à atteindre par notre approche de recherche peuvent être résumés de la manière suivante:

1. Développer un schéma de subdivision volumétrique permettant d'obtenir des surfaces lisses aux frontières;

2. Permettre l'obtention d'informations analytiques pour toutes positions arbitraires sur le maillage des frontières via un algorithme de projection de subdivision de limite;
3. Améliorer l'exactitude des maillages raffinés et adaptés par l'utilisation des limites de subdivision de Loop;
4. Maintenir une continuité adéquate pour les maillages de frontière générés par les subdivisions adaptatives;
5. Générer, par des subdivisions adaptatives, des maillages volumétriques non dégénérés dans le contexte de simulations numériques;
6. Intégrer en un seul schéma, par une méthode de raffinement à un seul niveau, l'adaptation de maillage et les techniques de subdivision;
7. Générer efficacement, par une méthode de raffinement à un seul niveau, des représentations multi-résolution de subdivision tout en préservant les caractéristiques géométriques spécifiques;
8. Permettre une l'application des développements apportés à d'autres types de subdivision de maillages.

La thèse elle-même se présente de la manière suivante. Le chapitre 2, « State of the Art » constitue une revue de la littérature concentrée sur les concepts de base des techniques de subdivision, les propriétés principales des opérateurs de subdivision, le schéma détaillé des surfaces de subdivision de Loop et les techniques de paramétrisation des surfaces de subdivision, particulièrement celle de Stam concernant l'évaluation exacte des surfaces de subdivision de Loop. Par la suite, nous présentons différentes représentations géométriques de modélisation de solides, étudions les avantages des techniques de subdivision des solides ainsi que les plus récentes approches des schémas de subdivision des solides. Une troisième partie fait le point sur les recherches reliées à la modélisation multi-résolution de subdivision selon trois catégories: la simplification des maillages, le raffinement des maillages et le contrôle des LODs (niveau de détails). Ces LODs sont une façon de représenter un maillage avec plusieurs niveaux de détails, chaque niveau étant de moins en moins complexe. La partie suivante couvre l'étude des possibilités d'utilisation des surfaces de subdivision multi-résolution à des fins d'ingénierie et présente également des travaux récents issus d'autres approches. Finalement, nous réaffirmons

notre proposition de recherche afin de résoudre les problèmes les plus courants liés à la mise en place des subdivisions solides, la paramétrisation des surfaces de subdivision et les raffinements adaptatifs.

Dans le chapitre 3 « Solid Subdivsion Based on the Loop Scheme » nous nous intéressons tout d'abord à la division topologique 1-à-8, au lissage géométrique basé sur la subdivision de Loop avec des règles préservant les caractères aigus (sommet fixe et pli d'arête) sur les surfaces de frontière des maillages tétraédriques. Nous discutons ensuite des difficultés liées à l'utilisation de la subdivision solide de Loop, surtout en ce qui concerne les structures de données et l'évaluation de la qualité des maillages. Cela démontre que les subdivisions uniformes et les itérations non-projetées à leur position limite ne rencontrent pas de manière évidente les besoins de précision et d'efficacité. Ensuite, nous nous intéressons aux subdivisions adaptatives qui présentent l'avantage de pouvoir éviter les problèmes cités plus haut. En terminant ce chapitre, nous discutons des difficultés d'intégration des raffinements adaptifs dans le modèle.

Dans le chapitre 4 « Adaptive Subdivision-based Refinements », nous présentons la méthode de raffinement à un seul niveau en la comparant avec la méthode de raffinement à niveaux multiples. La méthode à un seul niveau est présentée afin d'unifier naturellement les différences topologiques pour l'ensemble des maillages de subdivision et pour le lissage géométrique sur les frontières. La mise en place d'une méthode de raffinement à un seul niveau sur des courbes, des surfaces triangulaires et des maillages tétraédriques est abordée dans les trois sous-sections suivantes.

Dans le chapitre 5 « Subdivision Limit Calculations on Irregular Surface Meshes », nous présentons premièrement deux méthodes de paramétrisation de subdivision de surface: la méthode exacte de Stam combinée avec la technique du miroir d'une part et d'autre part la méthode d'évaluation de Persson qui permet de pallier aux limitations des méthodes existantes dont nous avons discuté dans le chapitre 2. Ensuite, nous démontrons les résultats obtenus en appliquant les définitions de plis et les subdivisions adaptatives à différents exemples représentatifs. Finalement, nous analysons et validons les résultats obtenues en mesurant la qualité des maillages et leur coût de calcul.

Le chapitre 6, « Conclusion and Perspectives » résume notre contribution et énumère aussi des idées de sujets futurs à étudier.

2. Revue bibliographique

Dans le but d'étudier la possibilité d'utiliser les limites de subdivision multi-résolution de Loop en tant que représentation géométrique pour unifier la modélisation, le maillage, l'analyse, le raffinement et la visualisation de simulations numériques, nous situons notre recherche dans un contexte plus large : les techniques de subdivisions, les techniques de subdivision pour la modélisation solide, les techniques de multi-résolution et le contrôle du niveau de détail de maillage (LODs).

2.1 La base des représentations multi-résolution de subdivision

L'idée de base sous-jacente aux représentations de subdivision est d'appliquer une subdivision récursive sur des maillages de contrôle grossiers pour obtenir des représentations lisses (Zorin, Schroder et al. 2000). Le processus de subdivision itératif est une séquence de raffinements à l'infini, où les niveaux d'itération sont décidés par des exigences concrètes sur la finesse et la régularité des maillages de subdivision. La connexion entre les niveaux de subdivision est réalisée en appliquant les opérateurs de subdivision *S* sur les maillages de subdivision M^j (voir Équation 1), dont les définitions sont basées sur les règles de subdivisions correspondantes. Dans certaines applications spécifiques, les opérateurs sont présentés comme des matrices de subdivision ou masques de subdivision.

$$M^{j+1} = S \cdot M^j, \ (j \in N). \tag{1}$$

2.1.1 Termes et définitions de base

Les représentations de subdivision obtenues en appliquant les règles de subdivision une seule fois est un maillage de subdivision M^j dont la densité est reliée au niveau de subdivision correspondant j. Évidemment, les maillages de subdivisions multiples avec résolutions diverses sont générés lors de séquences de subdivision infinie approchant les limites de subdivision L (voir Figure 1)

Figure 1: Illustration d'une séquence de subdivision infinie.

Ces maillages de subdivisions multiples sont le résultat du raffinement du maillage de contrôle initial, chacun apportant une approximation de la forme géométrique initiale au niveau de résolution spécifié. Donc, les techniques de subdivision supportent naturellement la génération de maillages de résolutions variées, où le niveau de détail (LODs) et la densité de chaque maillage de subdivision sont distribués uniformément. Les représentations multi-résolution proposées dans notre recherche sont composées de plusieurs parties de maillage dont les niveaux de détails locaux sont différents ou non-uniformes.

Notre travail de raffinement de subdivision implique des maillages multi-résolution à une, deux et trois dimensions : courbes, maillages de surface triangulaire et maillages tétraédriques.

2.1.2 Propriétés des opérateurs de subdivision

Habituellement l'opérateur de subdivision S est une combinaison entre deux sous-fonctions : la fonction de scission topologique $Split()$ et la fonction de lissage géométrique $Smoothing()$ (Warren and Weimer 2002). Lors d'opérations réelles, cette combinaison peut être comprise comme étant constituée de deux sous-processus : 1) insertion uniforme de nouveaux sommets (sommets impairs) entre les sommets existants (sommets pairs) et 2) lissage géométrique de tous les sommets en effectuant une mise à jour de leur position.

Il existe deux types de schémas de subdivision : par approximation et par interpolation. La distinction entre ces deux schémas est faite en vérifiant si les positions géométriques des sommets de contrôle existants ont été changées lors du lissage géométrique. Pour les surfaces triangulaires, les deux principaux schémas de subdivision sont : la subdivision de Loop (un schéma de subdivision d'approximation) et la subdivision Modified Butterfly (un schéma de subdivision d'interpolation). Habituellement, les nouveaux maillages de subdivision générés sont rétrécis vers l'intérieur des maillages de contrôle initiaux après avoir appliqué de manière itérative un schéma de subdivision d'approximation. Pour les maillages de subdivision par interpolation, les maillages conservent la même dimension que les maillages initiaux de contrôle. Comme les deux schémas impliquent la sous-fonction $Split()$, ils peuvent être classés comme schémas de bissection d'arêtes.

Dans les travaux de Zorin, Schroder et al. (Zorin, Schroder et al. 1997, Zorin, Schroder et al. 2000, Schroder 2004), les caractéristiques principales des propriétés des opérateurs de subdivision S peuvent être résumés de la manière suivante:

Support des fonctions. Le point de contrôle initial a une influence relativement restreinte sur la forme des représentations de subdivision finales (voir Équation 2).

$$\exists l_0, l_1 \in N \Rightarrow \forall k : s_{i,k} = 0, i \notin \{2k - l_0, 2k + l_1\}, \ (i \in N). \tag{2}$$

Définition locale. Le calcul de la position d'un nouveau sommet dépend seulement des sommets de contrôle initiaux voisins (voir Équation 3).

$$v_i^{j+1} = \sum_k s_{i-2k,k} \cdot v_k^j, \ (i, j \in N). \tag{3}$$

Invariance affine. Les transformations, comme la translation et la rotation des sommets de contrôle initiaux, sont conservées de manière constante durant la subdivision (voir Équation 4).

$$a \cdot v_i^{j+1} + \vec{b} = \sum_k s_{i,k} (a \cdot v_k^j + \vec{b}), \ (\vec{d} \in R^d, a \in R). \tag{4}$$

Symétrie des indices. Les matrices de subdivision sont indexées de manière symétrique (voir Équation 5).

$$\begin{aligned} s_{i,k} &= s_{i+2,k+1}, \\ s_{2k-l,k} &= s_{2k+l,k}, \end{aligned} \quad (i,k,l \in N). \tag{5}$$

Ces quatre propriétés des opérateurs de subdivision constituent les bases des méthodes de subdivision qui les rendent compétitives afin de répondre aux problèmes informatiques et topologiques que les utilisateurs de modélisation géométrique rencontrent. Tout d'abord, le support des fonctions est particulièrement favorable à la réalisation de raffinements de subdivision adaptatifs efficaces. En effet, lorsque quelques sommets de contrôle d'un maillage sont modifiés, il faut seulement mettre à jour les informations des sommets influencés localement et non pas tout le maillage (Wang 2006).

Deuxièmement, en ce qui concerne les propriétés de définitions locales, notons qu'elles peuvent être calculées rapidement. La raison en est que lors de l'insertion de nouveaux sommets sur le maillage raffiné, seuls les sommets voisins doivent être utilisés pour obtenir les positions des sommets insérés. Troisièmement, la propriété d'invariance affine est essentielle afin de supporter une représentation géométrique unifiée dans le contexte des raffinements multi-résolution. Les propriétés de symétrie des indices rendent le calcul des matrices de subdivision non seulement possibles mais aussi rapides, ce qui supporte directement une évaluation efficace des maillages de subdivision nouvellement générés. En résumé, toutes ces caractéristiques sont utiles non seulement pour l'analyse des matrices de subdivision mais aussi pour supporter la continuité et la convergence des limites de subdivision (Zorin, Schroder et al. 2000, Wang 2006).

2.1.3 Le schéma de subdivision de Loop

Dans le cadre de notre recherche, l'opérateur de subdivision *S* est basé sur le schéma de Loop, qui a été proposé par Charlie Loop en 1987 (Loop 1987). Les règles de subdivision pour le schéma de Loop sont spécifiées pour deux surfaces triangulaires distinctes : 1) intérieur et 2) frontière ou pli. Pour chaque partie de la surface, les masques de subdivision des nouveaux sommets impairs et des sommets pairs existants sont définis respectivement. De manière générale, pour les surfaces de subdivision triangulaires, les sommets intérieurs avec une valence de 6 et les sommets frontières avec une valence de 4 sont des surfaces régulières.

Les règles de subdivision de Loop jouent un rôle fondamental dans la création d'une représentation multi-résolution de subdivision unifiées pour les tâches de raffinement et ce non seulement pour la modélisation des surfaces mais aussi pour les subdivisions des solides.

2.1.4 Limites de la subdivision de Loop

Dans les recherches de Loop, le schéma de subdivision basé sur les splines quadratiques a été développé pour générer une surface lisse à partir d'un maillage de contrôle triangulaire grossier avec une topologie arbitraire en utilisant un nombre infini de subdivisions récursives. Habituellement, les surfaces de subdivision aux limites de Loop ont une continuité de surface C^2 partout sauf pour les parties près des sommets extraordinaires, des frontières et des plis où seulement une continuité de surface C^1 est assurée (Zorin, Schroder et al. 2000). Cela signifie que les dérivées secondes de la surface limite de la subdivision de Loop sont continues dans presque tous les cas. Cette propriété de continuité répond à un facteur important pour l'analyse

d'ingénierie surtout en ce qui a trait à l'analyse des éléments finis (FEA) (Burkhart, Hamann et al. 2010). Pour nos raffinements de subdivision, nous utilisons cette limite de subdivision afin de représenter la véritable géométrie de l'objet physique modélisé.

Les techniques de paramétrisation sur les surfaces de subdivision (Stam 1998, Stam 1998) ont établi de manière directe un lien entre le maillage de contrôle initial grossier et ses limites de subdivision. Le niveau de difficulté de la paramétrisation des limites de subdivision dépend de la complexité topologique des surfaces locales évaluées. Vers 1992, Lai a développé un algorithme de calcul des matrices B-nets (Lai 1992) qui permet de paramétrer topologiquement de manière directe un maillage triangulaire régulier en utilisant les patchs de Bézier qui sont dérivés d'une collection de box-splines complexes à deux variables.

Un peu plus tard, en 1998, Jos Stam dans sa recherche (Stam 1998b) a identifié des modèles de maillages topologiquement réguliers et triangulaires comme une série de patchs triangulaires réguliers. En pratique, cette matrice d'évaluation peut être utilisée directement afin de calculer la position limite de tout emplacement arbitraire sur le maillage triangulaire régulier. La contribution importante de Jos Stam a été de développer un algorithme d'évaluation qui permet la paramétrisation des maillages de surface de subdivision triangulaire ayant une topologie irrégulière, soit une facette triangulaire possédant un sommet extraordinaire.

2.2 Subdivision des solides

Le concept de base des schémas de subdivision provient de l'algorithme simplifié de Chaikin qui permet de générer des courbes lisses arbitraires à partir d'un nombre limité de points de contrôle (Chaikin 1974). La subdivision cubique de Catmull-Clark (Catmull and Clark 1978) et la subdivision quadratique de Doo-Sabin (Doo and Sabin 1978) ont élargi le développement des schémas de subdivision des courbes aux surfaces. Aux alentours de 1998, pour la première fois les surfaces de subdivision ont été utilisées pour la modélisation des personnages du film d'animation « Geri's Game » (Kerlow 2004). Depuis, les représentations utilisant les subdivisions ont gagné en popularité pour représenter des surfaces lisses avec des topologies arbitraires. Par contre, jusqu'à maintenant, les implications des techniques de subdivision pour la modélisation des solides n'ont pas encore été beaucoup étudiées.

2.2.1 Les représentations courantes des modèles solides

Il y a trois méthodes courantes pour représenter les géométries solides : la géométrie constructive solide (CSG), la représentation de frontière (B-rep) et la décomposition cellulaire (Rossignac and Requicha 1999). Chaque représentation géométrique de solide possède ses propres structures de données, opérations mathématiques et algorithmes de calcul.

Dans notre recherche, nous proposons d'utiliser la subdivision des solides comme représentations géométriques des solides. Une subdivision solide peut être définie comme étant la limite des séquences de raffinement successif des maillages volumiques. Plus spécifiquement, la subdivision d'un solide tétraédrique est constituée de quatre éléments de base : sommet, arête, face et tétraèdre. Les informations topologiques de subdivision des solides sont le lien entre ces éléments de base. Les subdivisions itératives des solides mettent à jour la connectivité des maillages mais lissent également leurs frontières. Tout comme la subdivision des surfaces, les opérations de subdivision des solides définissent les relations paramétriques entre les sommets de contrôle initiaux et les sommets de contrôle des subdivisions des niveaux supérieurs des frontières des solides. Donc, la subdivision des solides fusionne les aspects positifs des représentations générées par les représentations B-reps et Octree.

2.2.2 Subdivision des solides

Jusqu’à maintenant, la plupart des travaux concernant les techniques de subdivision ont été développées pour modéliser des surfaces avec des maillages. La première recherche sur les subdivisions de solides a été présentée dans les travaux de MacCracken et Joy (MacCracken and Joy 1996). Leurs travaux utilisaient une extension du tenseur de produit de subdivision de Catmull-Clark qui raffine successivement les maillages hexaédriques tridimensionnels. Cette méthode de raffinement de subdivision Catmull-Clark, très innovatrice, a dirigé l’attention vers les objets solides déformés ayant une topologie arbitraire en implantant des techniques de subdivision.

Ce sont les travaux de Bajaj et al. (Bajaj, Schaefer et al. 2002) et de Chang et al. (Chang, McDonnell et al. 2002) qui ont ouvert de nouvelles voies pour les schémas de subdivision de solides pour la modélisation de manière libre. Dans leur publication de 2002, Chang et al. (Chang, McDonnell et al. 2002) présentent un schéma de subdivision de solides approximatif en utilisant les box-splines à trois variables afin de subdiviser les maillages tétraédriques. Toutefois,

la complexité de ces règles box-splines à trois variables entrave l'utilisation pour l'approximation de maillages tétraédriques lisses. De plus, cette recherche implique la génération d'octaèdres à l'intérieur d'un maillage régulier de tétraèdres, ce qui limite leur utilisation pour les maillages de volumes ayant une topologie complexe.

En 2003, Chang et al. (Chang, McDonnell et al. 2003) ont proposé un schéma d'insertion de subdivision de solides qui utilise une combinaison d'interpolations simple linéaires pour générer des limites de maillages quadrilatéraux. La continuité de la surface limite est égale à C^1. Tous ces premiers travaux sur la subdivision des solides étaient proposés comme méthodes pour générer des régions tridimensionnelles lisses déformables qui ne peuvent toutefois être utilisés pour la discrétisation adaptative.

Récemment, Burkhart et al. (Burkhart, Hamann et al. 2010) ont présenté un schéma approximatif de subdivision, qui permet de raffiner de manière adaptative les maillages tétraédriques. Leur schéma de subdivision d'un solide implique un ratio de division de 1 à 4 pour les faces et les arêtes et un lissage géométrique de schéma de subdivision $\sqrt{3}$. Ce schéma a été développé par Leif Kobbelt (Kobbelt 2000). Tout comme la subdivision de Loop, la subdivision $\sqrt{3}$ peut être utilisée pour les maillages de surface triangulaire et la continuité de la surface limite est de C^2 presque partout. Les résultats de subdivision de Burkhart et al. sont obtenus en combinant les schémas de subdivision $\sqrt{3}$ et les critères de raffinement adaptatifs sans approcher répétitivement des limites de subdivision. Cet aspect rend leur lissage géométrique plus près des fonctions d'approximation que du véritable lissage de subdivision. Pour des applications réelles, les maillages tétraédriques complexes ont deux problèmes importants : la qualité du maillage et l'efficacité du raffinement adaptif. Les travaux de Burkhart et al. seraient plus avantageux si ces deux sujets y étaient discutés.

Afin de pouvoir utiliser de manière optimale les subdivisions de solides pour la modélisation, il est essentiel d'utiliser des subdivisions adaptatives. Leur utilisation dans des applications réelles implique la création de maillages multi-résolution, ce qui suppose des recherches sur la simplification des maillages, le raffinement des maillages et le contrôle des LODs.

2.3 Modélisation multi-résolution par subdivision

Le principe de base de la modélisation multi-résolution débute avec la présentation des modèles géométriques hiérarchiques pour les algorithmes de surfaces visibles (Clark 1976). Cette base présentait une tentative d'utilisation des modèles hiérarchiques dans un cadre général afin de fournir différents niveaux de détails dans une scène avec des cadres cohérents.

La plupart des travaux de modélisation multi-résolution actuels touchent la modélisation de maillages de surface polygonale. De manière générale, ceux-ci peuvent être classés en deux catégories : simplification des maillages et raffinement des maillages.

La simplification des maillages est faite en créant de manière approximative plusieurs maillages simplifiés comportant moins de détails à partir d'un modèle de maillage initial massif en utilisant des algorithmes de simplification. La simplification peut être vue comme un processus d'optimisation sous contraintes à partir de la mise en place d'opérateurs globaux ou locaux où la fidélité ou le budget triangulaire sont utilisés comme contraintes (Luebke, Reddy et al. 2002).

Le raffinement de maillages a pour but d'améliorer la résolution et d'introduire plus de détails dans un maillage initial grossier en utilisant des algorithmes de raffinement. Les surfaces de subdivision en tant que méthode de raffinement efficace lorsque combinée avec des structures multi-résolution sont très utiles afin de représenter des maillages massifs avec une topologie arbitraire complexe.

Le niveau de détails (LODs) fait référence à l'exactitude d'un maillage pour un modèle géométrique (De Floriani et Magillo 2002). L'histoire des recherches sur le niveau de détails suit en parallèle le développement de la modélisation multi-résolution. LODs est habituellement utilisé afin d'effectuer le rendu d'objets distants en simplifiant le niveau de détails. Les premières applications et les plus récentes approches sur le niveau de détails ont été résumées par Luebke, Reddy et al. (Luebke, Reddy et al. 2002). Dans les plus récentes recherches, les LODs sont souvent utilisés afin de rendre les modèles de terrain plus interactifs.

2.4 Les applications d'ingénierie des techniques de subdivision multi-résolution

Depuis le succès obtenu par l'utilisation de la subdivision de surfaces afin d'animer des personnages de films, les techniques de subdivision multi-résolution (MRS) ont été de plus en plus intégrées dans différents logiciels, par exemple pour l'infographie et les reconstructions de surface dans le domaine biomédical. Toutefois, malgré leur popularité et leur développement croissants, les avantages de l'utilisation des techniques MRS n'ont pas été pleinement étudiés pour les logiciels d'ingénierie. De plus, parmi les rares contributions du MRS au design par simulation, les approches ont été limitées à la modélisation de surface et n'ont pas été utilisées pour la modélisation volumétrique.

Vers 2001, Gonsor et Neamtu ont étudié les possibilités d'utilisation des surfaces de subdivision dans le système de modélisation géométrique de surface de Boeing (the Aero Grid and Panel System (AGPS)) (Gonsor et Neamtu 2001). Dans ce rapport, les auteurs identifient les exigences essentielles pour l'utilisation de la subdivision selon une perspective d'ingénierie : la possibilité d'interpolation et la continuité des plans tangents.

En ce qui concerne les plus récentes approches sur les applications d'ingénierie de l'utilisation de la subdivision de surface, nous devons mentionner également les recherches de Cirak et al. (Cirak, Ortiz et al. 2000), ceux de Persson et al. (Persson, Aftosmis et al. 2006), de Ito (Ito, Shih et al. 2009) et de Burkhart et al. (Burkhart, Hamann et al. 2010) où les deux premiers travaux sont des ressources pratiques pour l'utilisation des techniques de paramétrisation de subdivision.

2.5 Rappel de notre sujet de recherche

La revue de la littérature faite ci-dessus montre d'une part que les représentations par subdivision ne sont pas seulement utilisées pour représenter la géométrie sous-jacente des modèles physiques complexes avec une topologie arbitraire mais peuvent aussi supporter les opérations de raffinement de maillages de différentes dimensions tant pour les maillages de surface que pour les maillages de volume. La combinaison de ces deux facteurs rend les représentations par subdivision très avantageuses pour l'unification du maillage et de la géométrie pour la simulation numérique dans le contexte de l'analyse d'ingénierie. D'autre part, ces recherches montrent que

les techniques de subdivision actuelles ont leurs limites et des problèmes non résolus demeurent concernant les paramétrisations des surfaces, la modélisation des solides, les subdivisions adaptatives dans le contexte de multi-résolution et l'intégration dans les logiciels d'ingénierie.

2.5.1 Rappel des problèmes

De ce que nous avons vu jusqu'ici, il est évident que le schéma de subdivision de Loop a bien été intégré à la modélisation de surfaces limites lisses ayant une continuité C^2. Toutefois, pour la modélisation d'un solide, notre recherche démontre que l'intégration n'a pas été complètement réalisée. De plus, le schéma de Loop constitue un schéma de subdivision d'approximation. Ce fait limite l'intégration dans les méthodes d'analyse numériques actuelles.

Tel que discuté dans la section 2.1.3, les limites de subdivision peuvent être calculées directement en utilisant les techniques de paramétrisation de subdivision de surface. En pratique, la matrice d'évaluation de Stam (Stam 1998b) est utilisée afin de calculer les positions limites des sommets arbitraires sur les surfaces de subdivision de Loop. Toutefois, le triangle où les sommets de surface sont calculés est limité à un seul sommet extraordinaire. Cette limitation indique de plus que la méthode d'évaluation de Stam ne peut calculer les limites de subdivision lorsque les maillages de contrôle incluent des facettes triangulaires avec des définitions de pli.

L'introduction d'un modèle multi-résolution permet de clarifier le fait que le maillage de subdivision multi-résolution est en fait un ensemble de parties de maillage multiples dont les LODs varient. Habituellement, cette sorte de maillage requière la méthode à plusieurs niveaux afin de stocker des parties de maillage de différents LODs. Cette méthode à plusieurs niveaux est basée sur le fait que la densité des différents maillages de subdivision peut être obtenue en appliquant de manière correspondante des raffinements de subdivision un certain nombre de fois. Donc, un maillage de subdivision multi-résolution peut être reconstruit en réunissant les parties des maillages de subdivision à plusieurs niveaux de subdivision. Toutefois, lors d'une utilisation réelle, cette méthode de génération des maillages multi-résolution est confrontée à plusieurs problèmes pratiques : 1) l'information requise afin d'accéder aux différents maillages de subdivision des différents niveaux doit être stockée. Ce stockage peut toutefois être lourd, surtout lorsque les maillages de subdivision sont des maillages volumétriques. 2) relier les différentes parties des maillages peut être ardu car il est difficile de lier la topologie des frontières des différentes parties de différents maillages ayant plusieurs LODs. Ce problème devient encore plus

complexe lorsque les parties des différents maillages sont des maillages de subdivision volumétriques.

2.5.2 Réaffirmation de nos travaux de raffinement

Pour notre recherche, nous proposons une utilisation des représentations par subdivision basée sur les limites de subdivision multi-résolution de Loop, ce qui implique fondamentalement trois techniques : le schéma de subdivision solide de Loop, les limites des subdivisions de Loop et la génération de maillages multi-résolution.

Premièrement, un nouveau schéma de subdivision de Loop basé sur la subdivision des solides est développé dans le cadre des maillages tétraédriques raffinés. Cette subdivision des solides peut être utilisée afin de générer des maillages tétraédriques ayant une surface frontière lisse. Notre règle de subdivision des solides est utilisée afin de mettre à jour les sommets existants et les nouveaux sommets insérés à l'intérieur des volumes et des surfaces frontières tout en préservant les arêtes vives et les sommets pointus. La connectivité des maillages tétraédriques sera particulièrement étudiée car elle constitue un facteur important influençant la complexité des algorithmes. De plus, la qualité des nouveaux maillages de subdivision tétraédriques générés sera aussi évaluée.

Deuxièmement, les techniques de paramétrisation des surfaces de subdivision seront abordées. Nous utiliserons ces dernières afin de projeter les nouveaux sommets sur les surfaces frontières limites. Les limitations des travaux de Stam vont être révisées dans notre recherche.

En troisième lieu, nous présenterons une méthode de raffinement à un seul niveau qui a été développée pour les raffinements adaptatifs et les évaluations précises des surfaces de subdivision de frontières. Cette méthode est introduite spécifiquement pour éviter les inconvénients des subdivisions uniformes et du stockage des informations hiérarchiques des raffinements non-uniformes traditionnelles à plusieurs niveaux. De plus, nous démontrerons que notre méthode de raffinement à un seul niveau peut être utilisée pour raffiner des courbes, des surfaces et des volumes.

Ensuite, nous présenterons les algorithmes utilisés pour la génération de maillages de subdivision multi-résolution, particulièrement en ce qui a trait à la génération de maillages de subdivision tétraédriques. Nous montrerons également des résultats de maillages raffinés obtenu

en appliquant des critères de subdivision analytiques extraits d'un contexte de simulation numérique.

Finalement, nous présenterons des exemples de maillages tétraédriques raffinés de façon adaptive contenant des plis de sommets et d'arêtes. Également, nous analyserons les performances de nos algorithmes de connectivité, de division de tétraèdres et de calcul de position limite.

3. Subdivision de solides avec le schéma de Loop

La méthode de raffinement par subdivision proposée dans notre recherche est composée de deux éléments essentiels : subdivision et adaptivité. Cette méthode peut être utilisée afin de raffiner les courbes, les surfaces de maillage triangulaires et les maillages volumétriques tétraédriques. Tels que présentés dans notre deuxième chapitre, les défis touchant l'intégration des techniques de subdivision sont liés à la modélisation des maillages volumétriques. En simulations numériques, les maillages volumétriques tétraédriques sont souvent utilisés pour des analyses d'ingénierie. C'est pourquoi nous nous concentrerons sur le développement des subdivisions de solides.

3.1 Schéma de subdivision de Loop

Un maillage volumétrique peut être vu comme étant constitué de deux parties : la frontière et l'intérieur. Pour un maillage tétraédrique, la frontière est une coquille surfacique composée de facettes triangulaires alors que l'intérieur est rempli de tétraèdres. Les subdivisions de Loop sont basées sur un schéma de bissection des arêtes. La subdivision complète d'un tétraèdre est réalisée en insérant un nouveau sommet au milieu de chaque arête, ce qui créé donc six nouveaux sommets. Ensuite, il faut connecter ces nouveaux sommets pour former des tétraèdres. Trois connectivités sont possibles. Afin d'obtenir une qualité de maillage appropriée, suite à cette subdivision, nous avons choisi une connectivité qui utilise le nouveau sommet qui engendre l'arête la plus courte. Le fractionnement d'un tétraèdre complet va donc créer huit tétraèdres plus petits et sera représenté par le fractionnement 1-à-8.

Le processus appliquant de manière itérative le schéma par subdivision de Loop sur un maillage tétraédrique de Schoenhardt est observé. De ce processus, nous remarquons que si le niveau de subdivision est augmenté, les frontières des maillages subdivisés correspondants tendent vers une surface lisse et la forme de cette surface rétrécit vers l'intérieur de la frontière du

maillage de contrôle initial. De plus, la densité du maillage augmente de manière égale. Ces mises à jour viennent du fractionnement topologique et du lissage géométrique.

Le lissage géométrique consiste en la mise à jour des positions géométriques 1) des sommets des tétraèdres existants et 2) des sommets insérés au milieu des arêtes. Ces règles de lissage géométrique peuvent de plus être classifiées selon qu'elles sont appliquées a) à des sommets sur une surface frontière et b) à des sommets à l'intérieur du volume. Nous pouvons classer ces règles en quatre différentes catégories : (1a) les sommets existants sur la surface frontière; (1b) les sommets existants à l'intérieur du volume; (2a) les nouveaux sommets insérés sur la surface frontière; (2b) les nouveaux sommets insérés à l'intérieur du volume. Pour notre recherche, afin d'obtenir un maillage lisse, nous nous attardons au calcul de la position de ces quatre catégories de sommets.

Au fur et à mesure que les niveaux de subdivision augmentent, les surfaces frontières des maillages tétraédriques deviennent graduellement plus lisses. Pour la modélisation des solides, afin de pouvoir se conformer aux critères physiques des modèles CAD, les applications véritables exigent que les caractéristiques aiguës locales des surfaces frontières, comme des courbures prononcées ou les arêtes vives, soient conservées durant les itérations de lissage géométrique. Afin de pouvoir rencontrer ces exigences, nous présentons une nouvelle formulation qui nous permet d'intégrer la création de pli de frontière dans le schéma de subdivision de solides de Loop.

Dans le présent travail, nous présentons deux types de création de plis de frontières : 1) sommet aigu et 2) arête vive. La création de sommet aigu consiste à maintenir leur position inchangée durant le lissage par subdivision. Les arêtes vives sont créées en marquant les arêtes des surfaces frontières des maillages volumiques comme des plis et à lisser ces arêtes ainsi marquées en tant que courbes isolées plutôt que d'appliquer les opérations de lissage de surface habituelle.

En comparaison avec le schéma de subdivision de solides à partir des Box splines des travaux de Chang (Chang, McDonnell et al. 2002), les avantages de notre approche peuvent être résumés ainsi : 1) la subdivision 1-à-8 ne génère que des tétraèdres alors que le subdivision topologique de Chang créé aussi des formes octaédriques. Notre méthode de subdivision topologique permet donc de simplifier les structures de données ce qui permet de travailler avec des maillages tétraédriques de topologie arbitraire. Cet aspect est également avantageux pour l'analyse

numérique car celle-ci ne tolère qu'un seul type d'éléments volumétriques pour la plupart de méthodes numériques; 2) nos algorithmes de lissage géométrique peuvent être divisés en quatre catégories de cas standards et une catégorie spéciale reliée aux sommets frontière ou aux arêtes vives. Notre but principal est ici d'étendre le schéma de subdivision de Loop du lissage des maillages de surface au lissage des maillages de surfaces de frontières de volume tétraédrique.

De plus, la fonction de conservation des caractéristiques aiguës de notre travail peut être naturellement intégrée aux procédures de lissage par subdivision. Cette intégration est un autre avantage de l'utilisation des représentations par subdivisions pour unifier la géométrie du lissage et la génération de maillages.

3.2 Les défis de la mise en œuvre du schéma de subdivision solide de Loop dans des applications réelles

Lors de nos travaux de subdivision solide, nous avons construit une structure de données permettant de garder la hiérarchie des maillages tétraédriques par subdivisions multiples selon diverses résolutions et de supporter la transmission tout au long des itérations des différentes subdivisions. Lors de notre subdivision itérative 1-à-8, un tétraèdre est subdivisé en huit tétraèdres plus petits et ceux-ci sont eux-mêmes divisés par la suite en huit autres tétraèdres. Les subdivisions se répètent jusqu'à ce que le maillage tétraédrique raffiné se rapproche de la géométrie requise. Nous utilisons un arbre à huit branches (voir eight-child tree, Wikipedia.org) afin de décrire cette structure hiérarchique liant tous les niveaux de subdivision et où un noyau de base représente le maillage tétraédrique initial de contrôle et les branches représentent les niveaux de subdivisions successives.

Puisque les maillages de subdivision de nos travaux serviront à des simulations numériques, il est important de s'assurer que les maillages créés lors des multiples itérations de subdivision ne sont pas dégradés. C'est pour cette raison que nous évaluons la qualité des tétraèdres générés en utilisant un moyen quantitatif : la mesure de forme des tétraèdres basée sur une transformation linéaire η (Dompierre, Vallet et al. 2005). D'après nos résultats d'évaluation, nous pouvons conclure que le fractionnement topologique 1-à-8 peut être utilisé pour préparer les maillages tétraédriques avec une valeur η valable pour les simulations numériques. Toutefois, le lissage géométrique des surfaces de frontière accroit la possibilité de générer des tétraèdres avec des formes non désirées. Également, les subdivisions uniformes ne sont pas utiles pour isoler ces

tétraèdres puisque les itérations de subdivision continuent uniformément. Tous ces facteurs expliquent pourquoi les maillages tétraédriques de subdivision sont dégradés.

Les facteurs défavorables à l'utilisation de la méthode traditionnelle de raffinement par subdivision uniforme sont les suivants : 1) elle est basée sur des subdivisions uniformes, ce qui est non seulement répétitif mais manque en plus de flexibilité. Elle ne permet donc pas de remplir des exigences pratiques comme le contrôle de la densité du maillage des volumes locaux ni d'isoler les parties de maillage non désirables afin de diminuer les possibilités de dégénérescence des maillages tétraédriques; 2) Il s'agit d'une méthode itérative d'approximation qui cherche à reproduire de manière approximative la géométrie véritable par des itérations de subdivision. Ce facteur peut potentiellement requérir plus de stockage et de ressources informatiques. Ce risque est aggravé lorsque nous utilisons l'arbre à huit branches afin de conserver des maillages de tétraèdres par subdivision à des niveaux différents où les éléments de subdivision sont dupliqués huit fois à chaque itération de subdivision.

Dans le but d'éliminer ces facteurs défavorables, nous avons intégré des subdivisions adaptatives dans nos travaux de raffinement. La subdivision adaptative permet de raffiner des parties du maillage au besoin plutôt que de réaliser des subdivisions uniformes intégrales. Toutefois, intégrer le support d'adaptabilité dans les subdivisions solides crée un autre problème : comment combiner ensemble les différentes parties de maillage ayant des résolutions différentes?

Ce problème est complexe sous deux aspects : 1) afin de pouvoir accéder la partie du maillage désirée, il faut faciliter la navigation à travers les différents niveaux de subdivision. Il s'en suit que les informations de connectivité de l'arbre à huit branches qui guide l'accès des plus hautes subdivisions aux plus basses et inversement (héritage des parents et rétrospection des enfants) doivent être gardées. Il est évident que ces informations sont encore plus encombrantes et compliquées, spécialement en ce qui concerne les maillages tétraédriques; 2) Afin de réunir les parties des maillages tétraédriques de différents niveaux, il faut travailler avec des tétraèdres ayant des angles faibles, comme les aiguilles, les tranches ou les éclats, ce qui nécessite un autre processus complexe de post-optimalisation (Shewchuk 1998; Volkov and Ling 2003). Finalement, ces deux aspects ne peuvent être résolus en utilisant seulement la méthode traditionnelle de raffinement par subdivisions qui combine le fractionnement topologique 1-à-8 et les raffinements uniformes par subdivision.

Afin d'améliorer la précision et l'efficacité des représentations par subdivision, nous avons intégré deux éléments essentiels dans la méthode de raffinement par subdivision : 1) les techniques de paramétrisation de subdivision de surface; 2) les raffinements adaptatifs. Afin de réunir ces deux éléments, nous avons développé une méthode de raffinement à un seul niveau. Cette méthode s'assure que les sommets surfaciques rencontrant les critères de raffinement sont projetés correctement à leur position limite. Cela génère éventuellement des maillages tétraédriques multi-résolution par subdivision composés de parties ayant des volumes différents à des résolutions différentes.

4. Raffinements adaptatifs par subdivision

Tel que discuté précédemment, les subdivisions adaptatives constituent une exigence afin d'améliorer la précision et l'efficacité lors de l'implantation du schéma de subdivision de solides de Loop dans des applications réelles. Dans le but de mieux interpréter les concepts derrière la méthode de raffinement par subdivision, nous présentons les mises en œuvre correspondant à trois types de représentations par subdivision : courbe, maillage de surface triangulaire et maillage tétraédrique.

4.1 Raffinements adaptatifs de courbe

Puisque les limites de subdivision présentent des propriétés intéressantes de continuité de surface et de géométrie analytique, nous avons utilisé les représentations par subdivision afin de décrire la géométrie véritable des modèles d'études. Pour les cas de raffinement de courbe, un maillage courbe par subdivision consiste en une série de sommets qui sont en fait les sommets des subdivisions projetés à leur position limite. De ce que nous avons écrit précédemment, il ressort que les raffinements uniformes ont comme inconvénients le manque de flexibilité pour le contrôle des niveaux concrets de détails. Dans notre recherche, nous avons modifié de manière adaptative les maillages courbes par subdivision tout en projetant directement les nouveaux sommets insérés à leur position limite. Les algorithmes supportant les raffinements adaptatifs de courbes sont constitués par les deux opérations suivantes : 1) insertion d'un nouveau sommet entre les arêtes sélectionnées du maillage raffiné 2) application des critères de raffinement. Les pseudo-codes des fonctions impliquées sont détaillés dans cette thèse.

En nous basant sur les résultats des maillages raffinés démontrés dans ce chapitre, nous pouvons conclure que notre méthode adaptative de raffinement de courbe peut être utilisée avec succès afin de raffiner de manière adaptative les maillages courbes. Les maillages courbes raffinés de façon adaptative sont placés exactement sur la limite de subdivision et leur précision peut être contrôlée selon des critères de raffinement.

Suite à cela, la méthode de raffinement à un seul niveau a été développée afin de faciliter l'intégration entre les techniques de subdivision et les raffinements adaptatifs. Les concepts de base derrière cette méthode à un seul niveau peuvent être également utilisés pour raffiner les surfaces et les maillages de volume.

Dans cette méthode à un seul niveau, la technique de paramétrisation des surfaces de subdivision est ajoutée afin que la position limite de n'importe quelle position arbitraire du maillage de contrôle initial puisse être obtenue. Cette paramétrisation est une application qui implique que les positions sur le maillage de contrôle initial aient leur position correspondante sur la limite de subdivision. En d'autres mots, chaque raffinement met à jour seulement la densité des parties de maillage sélectionnées et même après plusieurs raffinements, seuls deux maillages sont gardés dans le système : le maillage de contrôle initial et le maillage raffiné dont les informations détaillées sont stockées en tant que Maillage_1 et Maillage_2.

De manière spécifique, le Maillage_1 est une liste contenant les sommets de contrôle initiaux. Ce Maillage_1 demeure inchangé et fournit seulement des informations pour le Maillage_2. Le Maillage_2 est une liste de sommets, dont les positions se trouvent sur la limite de subdivision. Lorsque la subdivision adaptative est appliquée, les nouveaux sommets sont ajoutés à leur position limite de manière adaptative sur le Maillage_2. Donc le Maillage_2 est mis à jour directement tant que le raffinement continue. Le processus complet de raffinement ne nécessite donc que le Maillage_1 et le Maillage_2. Il n'est donc pas nécessaire d'avoir un grand nombre de maillages pour garder les informations hiérarchiques générées par les différentes itérations de subdivision.

De plus, entre le Maillage_1 et le Maillage_2 de la méthode à un seul niveau, les opérations de raffinement par subdivision jouent un rôle de projection. Selon les informations du Maillage_1, nous pouvons obtenir toutes les positions limites possibles des limites de subdivision. Le processus de raffinement vise à projeter les positions limites rencontrant les

critères de raffinement définis et à les insérer dans le Maillage_2. Nous présentons de plus dans notre recherche des modèles de fractionnement topologique différents afin de répondre aux problèmes de raccordement durant le raffinement adaptatif du maillage de surface ou de volume.

4.2 Raffinements adaptatifs sur les surfaces triangulaires

Lorsque le raffinement adaptatif est appliqué sur des maillages de surface ou de volume, l'élément de base pour le fractionnement devient des facettes triangulaires ou tétraédriques. Cette réalité indique que les topologies impliquées doivent être traitées de manière différente. Dans notre travail de raffinement adaptatif, les surfaces triangulaires sont traitées en deux parties : la frontière et l'intérieur. Cette frontière est une courbe liée à une collection de segments d'arête et l'intérieur est une surface intégrée de facettes triangulaires. Ce raffinement adaptatif de surface triangulaire peut être résumé de la manière suivante : 1) La courbe frontière est traitée comme une courbe de subdivision dont les segments d'arête sont scindés en deux de façon adaptative. Par la suite, les nouveaux sommets d'arête insérés sont projetés aux limites des frontières courbes. 2) Parallèlement, les facettes triangulaires intérieures sont divisées en quatre ou deux facettes triangulaires plus petites selon les informations marquées sur les arêtes des facettes correspondantes.

À chaque étape de raffinement, les facettes triangulaires rencontrant les critères de raffinement sont sélectionnées pour le fractionnement. Par la suite, les arêtes correspondantes aux facettes triangulaires qui se qualifient sont étiquetées comme « à fractionner ». Lorsque les informations de fractionnement des arêtes sont partagées avec les facettes triangulaires voisines, la vérification des informations de fractionnement mises à jour est essentielle non seulement après chaque étape de raffinement mais aussi durant les fractionnements de triangles. Pour éviter les déconnections abruptes ou les cassures sur les maillages triangulaires, nous subdivisions de nouveau toutes les facettes triangulaires dont les arêtes ont été étiquetés « à fractionner ». Selon le nombre d'arêtes étiquetés pour le fractionnement, les fractionnements de triangles sont classés en deux types de patrons : le fractionnement 1-à-2 et le fractionnement 1-à-4. De manière plus brève, les raffinements adaptatifs sur les surfaces triangulaires sont en fait des subdivisions d'arêtes. L'algorithme impliqué dans le raffinement adaptatif crée des surfaces triangulaires qui sont détaillées dans cette thèse.

4.3 Raffinements adaptatifs sur les maillages tétraèdres 3D

La complexité topologique et l'augmentation des données géométriques rendent le raffinement des maillages tétraédriques plus complexe. Les avantages à l'intégration naturelle des subdivisions de solides avec des raffinements adaptatifs par l'utilisation de la méthode à un seul niveau sont plus évidents lorsque les subdivisions adaptatives sont faites sur des maillages de tétraèdres, particulièrement ceux avec des volumes importants et des topologies arbitraires.

Ici, les volumes des maillages 3D sont traités en deux parties : les frontières et l'intérieur. Ces frontières forment une surface d'enveloppe intégrant les facettes triangulaires alors que l'intérieur est rempli de tétraèdres. Dans nos travaux de raffinements adaptatifs, les surfaces de frontières sont traitées comme des surfaces de subdivision, où les fractionnements topologiques de triangles et le lissage géométrique sont tous deux appliqués sur les facettes triangulaires sélectionnées.

La base de notre raffinement adaptatif est 1) sélectionner tous les blocs de subdivision fondamentaux rencontrant les critères prédéfinis de subdivision 2) étiqueter toutes les arêtes de fractionnement reliées, ce qui correspond à l'insertion d'un nouveau sommet sur les arêtes étiquetées 3) si les nouveaux sommets créés se trouvent sur les frontières, les masques d'évaluation seront appliqués afin de s'assurer qu'ils se trouvent à leur position limite. En pratique, si le critère de subdivision est basé sur l'évaluation d'arête comme la courbure ou la longueur d'arête, toutes les arêtes qualifiées seront immédiatement sélectionnées durant la première étape. De manière générale, les raffinements sur des maillages tétraédriques sont des subdivisions sur les arêtes, cependant l'élément de base qui est réellement subdivisé est le tétraèdre.

Tel que nous l'avons présenté plus tôt, pour un maillage tétraédrique, la manière de différencier la frontière et l'intérieur passe par la connectivité des tétraèdres. Dans le cadre de notre recherche, nous nous intéressons aux pseudo-codes reliés au calcul de la connectivité du maillage de tétraèdre. Une fois la connectivité établie, les opérations de lissage seront appliquées seulement sur les surfaces frontières alors que les fractionnements topologiques seront eux appliqués aux tétraèdres qualifiés. Habituellement, le fractionnement d'un tétraèdre est un fractionnement 1-à-8. Puisque les algorithmes de subdivision de nos travaux sont basés sur les arêtes dont les informations de fractionnement sont partagées par d'autres tétraèdres voisins, nous ajoutons donc deux autres types de fractionnement : le fractionnement 1-à-4 et le fractionnement

1-à-2 afin de fractionner tous les tétraèdres voisins des arêtes marquées « à fractionner ». Ceci permet d'éviter les déconnections abruptes et les cassures sur les maillages de tétraèdres raffinés de manière adaptative.

De manière générale, les subdivisions d'arêtes des maillages tétraédriques peuvent être catégorisées en deux parties : 1) à l'intérieur, tous les tétraèdres correspondants sont fractionnés en huit, en quatre ou en deux plus petits tétraèdres; 2) sur les surfaces des frontières, les faces triangulaires sont fractionnées en quatre ou en deux faces triangulaires plus petites. Au même moment, le lissage géométrique est appliqué aux nouveaux sommets d'arête sur les faces du fractionnement des frontières. En fait, ce lissage sert à projeter tout nouveau sommet de surface frontière à partir du maillage de contrôle initial jusqu'à sa position limite de subdivision en utilisant la technique d'évaluation de subdivision de Loop, dont les détails seront plus largement abordés plus loin. Ces deux parties combinées permettent que des maillages tétraédriques complets soient raffinés de façon adaptative alors que les surfaces frontières correspondantes sont lissées et gardées directement sur les limites de subdivision.

En résumé, l'essence du raffinement adaptatif est de combiner les fractionnements topologiques et le lissage géométrique qui sont définis comme des fractionnements d'arêtes et des projections de frontières. La méthode à un seul niveau est développée afin d'unifier naturellement ces deux opérations ce qui facilite par la suite l'intégration des subdivisions adaptatives.

5. Calcul de limite de surface de subdivision sur maillages irréguliers

Dans le chapitre 5, nous comparons les techniques existantes d'évaluation de subdivision de surface: la méthode d'évaluation de Joe Stam (Stam 1998b) combinée avec la technique de miroir proposée par Cirak et al. (Cirak, Ortiz et al. 2000) et la méthode simplifiée d'évaluation de Persson (Persson, Aftomis et al. 2006). Notre comparaison se concentre sur la démonstration de la faisabilité et l'efficacité des calculs des positions limites pour les maillages irréguliers.

5.1 La technique de miroir

Afin de pouvoir évaluer toute position arbitraire sur tous les types de surfaces triangulaires ayant une topologie arbitraire, nous avons intégré les techniques de miroir pour résoudre les problèmes

de plis. Cette technique fait référence à la thèse de Ph.D. de Schweitzer (Schweitzer 1996) laquelle a été citée par Cirak et al. pour calculer les positions de lissage sur les sommets de frontière des surfaces de subdivision. La technique de miroir peut être expliquée comme étant des sommets temporaires symétriquement copiés tout le long des rebords de frontière.

Dans nos travaux, nous traitons les rebords de plis définis en utilisant la technique de miroir. Nous déduisons la position des sommets externes à partir des sommets existants situés de l'autre côté du pli. Nous avons alors conclu que cette technique ne peut pas être utilisée en tout temps puisque qu'il est difficile d'établir le lien entre les points existants et les positions des sommets miroir pour les situations illustrées dans notre recherche.

5.2 La méthode de paramétrage de surface de subdivision de Persson et al.

La méthode de paramétrage de surface de subdivision de Persson et al. (Persson, Aftomis et al. 2006) découle de la théorie sous-jacente à la méthode d'évaluation exacte de Jos Stam. Dans la méthode de Persson, des subdivisions récursives sont utilisées pour transformer les facettes triangulaires irrégulières en sous-facettes triangulaires régulières. Ces facettes triangulaires régulières peuvent être évaluées en utilisant l'expression des matrices B-nets.

Contrairement à la méthode d'évaluation exacte de Jos Stam, Persson et al. proposent un schéma où toutes les positions arbitraires d'une facettes triangulaire irrégulière avec pli ou avec plus d'un sommet extraordinaire peuvent être paramétrisées à l'aide d'expressions des matrices B-nets.

Un traitement spécial doit être mis en place afin de stopper les subdivisions récursives lorsque la position de paramétrage est très proche d'un sommet extraordinaire ou d'un pli. Dans ces cas, la position de paramétrage doit être légèrement ajustée afin de se trouver au centre de la partie de la sous-facette triangulaire, laquelle partie deviendra une sous-facette triangulaire régulière au niveau de subdivisions suivant.

Dans les sections suivantes, nous présenterons plusieurs exemples prouvant l'importance des plis et nous démontrerons leur cohabitation avec le maillage adaptatif à un seul niveau unique. Des résultats de performances de la méthode de paramétrage de surface de subdivision de Persson et al. (Persson, Aftomis et al. 2006) sont également présentés.

6. Conclusion et perspectives

Dans cette thèse, nous avons exposé les grandes lignes d'une approche novatrice de recherche pour l'utilisation des représentations par subdivision permettant le raffinement adaptatif du maillage et la représentation géométrique à des fins d'analyse d'ingénierie. Cette approche incorpore à la fois les techniques de subdivision de limites pour les représentations géométriques et les subdivisions adaptatives pour la modification des maillages. Elle est constituée de quatre parties : i) le raffinement par subdivisions adaptatives via le fractionnement topologique par arêtes et le lissage géométrique des maillages des frontières; ii) l'établissement d'une méthode de raffinement à un seul niveau qui supporte la génération de maillages multi-résolution par subdivisions; iii) l'introduction de critères analytiques afin de guider le raffinement adaptatif; iv) l'évaluation de la qualité des nouveaux maillages tétraédriques générés et de l'efficacité du calcul de position.

Dans la section 6.1, nous présentons le résumé de nos contributions en décrivant les trois aspects essentiels qui sont les suivants: les innovations par subdivision de solides, les impacts de l'adaptation des maillages dans les simulations numériques et la comparaison des techniques de paramétrisation des surfaces par subdivision de Loop.

Nous avons par la suite conclu que notre recherche impliquait plusieurs champs de recherche très actifs : surfaces de subdivision, subdivision de solides, adaptation des maillages et modélisation multi-résolution. Dans la section 6.2, nous explorons trois approches possibles pour le futur : la subdivision inverse, l'évaluation de la qualité des maillages et le calcul de la connectivité du maillage.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS AND SYMBOLS

ADM	Attribute Deviation Metric
AGPS	Aero Grid and Panel System
AIF	Adjacency and Incidence Framework
B-rep	Boundary representation
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CFD	Computational Fluid Dynamics
CSD	Computational Structural Dynamics
CSG	Constructive Solid Geometry
FEA	Finite Element Analysis
LODs	Level Of Details
LSM	Level Set Method
MRA	Multi-Resolution Analysis
MRSS	Multi-Resolution Subdivision Surface
NURBS	Non-Uniform Rational Basis Splines
QEM	Quadric Error Metric
STL	STereoLithography

CHAPTER 1
INTRODUCTION

Simulating complex physical phenomena on real world objects is highly demanded in a broad range of engineering applications, such as physics-based animation, biomedical simulation and industrial design and manufacturing. Traditionally, an engineering analysis model comprises two main aspects: geometric representation of the computational domain and physical models in the form of Partial Differential Equations (PDEs). Precise geometric description of the objects and accurate modeling of the physical phenomena are thus two complementary research fields that come together to support numerical simulations.

1.1 An Overview of the Numerical Simulation Process for Engineering Analysis Purposes

In the context of engineering analysis, the general numerical simulation process can be presented in the following steps. This process begins with a physical object and phenomenon in the real world. The practical approach is firstly to create and discretize a physical model using engineering analysis and computational skills, secondly to construct the geometric representation satisfying the mathematical precision requirement defined in the previous step, thirdly, to generate a mesh based on the geometry definition, fourthly to solve the PDE using a solver until numerical analysis requirements are met, and finally to visualize and analyze the results (Owen 2005). In brief, these five steps involve contributions from multiple disciplines: physical modeling, geometric modeling, mesh generation, numerical analysis and computer visualization.

Traditionally, these five interconnected disciplines in real numerical simulation applications are treated as independent fields of research and each holds its own methods and representations. Under these circumstances, extra efforts are required to reconcile their incompatibilities, which may introduce inaccuracies during data transitions between them.

For example, the underlying geometry of simulated models in real applications is required to meet high fidelity criteria, so that a numerical problem can be accurately defined. More specifically, the whole simulation process relies on an adequate capacity in defining, representing

and possibly modifying the geometry. The procedure of finding a numerical solution assisted by mesh adaptation is detailed in Figure 1.1.

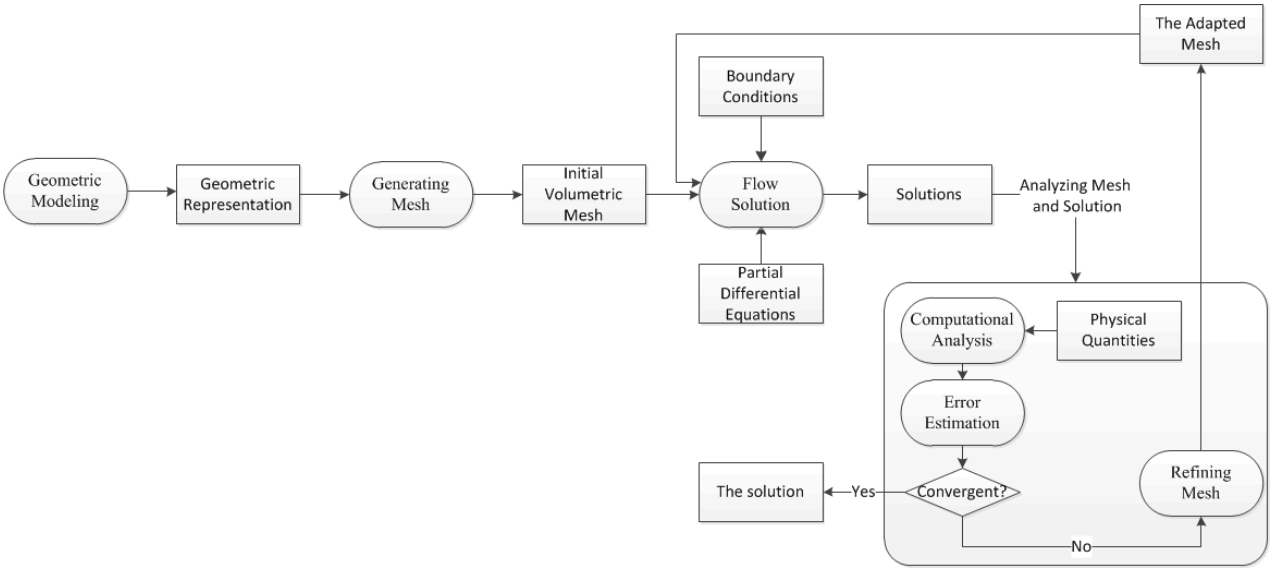

Figure 1.1: The detailed procedure of finding the numerical solution assisted by mesh adaptation.

From Figure 1.1, it can be observed that the underlying geometry of a physical model is initially defined by a geometric representation in geometric modeling. Then, to facilitate numerical analysis and computer visualization, a mesh is introduced in mesh generation for describing the original geometry definition. Next, during mesh adaptation, this mesh is adapted in an iterative loop, where the modified geometry information is interpreted in physics quantities for convergence analysis. Finally, this loop is terminated when the numerical solution meets certain criteria. Thus, in the whole procedure above, a mesh serves as ‘the geometry carrier’ to support numerical analysis. Here one question arises: whether and how this newly adapted mesh discretized in mesh adaptation is assured to conform to its initial geometry defined in geometric modeling.

For the purpose of improving the precision and efficiency of numerical simulations, we propose seeking a unifying geometric representation that serves as a common foundation that can be used throughout the whole simulation process, integrating both geometry and mesh to support mesh refinement and geometry modification.

1.2 Current Geometric Representation Approaches in Numerical Simulations

Geometric representation, as a fundamental element in geometric modeling, is used to specify distinct attributes and features of physical objects. Currently, two types of geometric representations: parametric boundary representation (B-rep) and polygon meshes, have been commonly integrated in numerical simulations for representing model geometry.

The B-rep approach consists in using parametric primitives, such as Non-Uniform Rational Basis splines (NURBS) or Bézier patches, to describe the continuous geometric shape of the computational domain boundaries with arbitrary precision (ISO 1994). During the numerical solution preparation, these continuous parametric primitives are discretized to a given level of accuracy, and the resulting discrete description is then used to construct the volume mesh, as a support for imposing boundary conditions and to solve the numerical problem.

While parametric B-rep has been proven very valuable for solving a large number of highly complex industrial problems (Labbé, Guibault et al. 2001, Corney, Hayes et al. 2005, Sunil, Agarwal et al. 2010), they suffer from important drawbacks especially when both a posteriori mesh adaptation and shape modification or shape evolution are part of the formulation for the numerical solution.

More specifically, parametric B-rep uses relatively few control parameters to construct models, which supports flexible operations for representing rich geometric features. One key advantage of using parametric B-rep lies in its capacity to manipulate or reconstruct parametric representations by interpolation or approximation. Besides, the geometry evaluation, such as curvature or derivative queries, is accessible on B-rep models. However, since the blending functions of B-rep parametric patches are generally based on tensor-products, the B-rep parameterization is most often limited to rectangular structures. This characteristic makes maintaining the continuity between the different patches or boundaries on B-rep models very difficult, particularly for geometric models of arbitrary topology or complex geometry. Under these circumstances, special treatments, such as Computer-Aided Design (CAD) repairs (Mezentsev and Woehler 1999, Yang and Han 2006), are introduced to amend the gaps, overlaps or intersections among parametric patches. But CAD repairs require developing expensive algorithms to deal with geometric errors due to tolerance problems (Mezentsev 2007).

Even though differences always exist between an actual object and its representation, it is essential to minimize errors introduced by successive transformations to the geometric models. Unfortunately, parametric B-reps usually need to be converted into a mesh for numerical analysis (Owen, White et al. 2002). This conversion is at the cost of losing original geometry information.

In the context of mesh adaptation near domain boundaries, on the one hand, the conformal relationship between the boundary mesh and the parametric B-rep primitives, sharp feature preservation (Qian and Zhang 2011) in particular, must be maintained. This disqualifies the purely discrete approach to domain representation. Furthermore, the adaptation operations, such as refinement or coarsening, on boundary meshes must account for the continuity definition of domain boundaries. Otherwise, significant discrepancies between the true domain shape and the adapted mesh may be introduced as the mesh is modified. On the other hand, adapting industrial models through a shape optimization process, particularly when volume meshes containing complex domain boundaries are used as geometric representations, often involves handling enormous amounts of information and updating mesh connectivity in real time, thereby imposing a huge burden to system resources. This burden challenges not only the system storage capacity, but also the computational rapidity of reconstructing the affected local regions on the adapted meshes.

Polygon meshes, as a second common geometric representation, are widely used for rendering geometric models with arbitrary topology in numerical simulations. For example, STereoLithography (STL) files frequently used as an alternative to parametric B-reps in 3D CAD systems are polygon meshes. The major difficulty of employing polygon meshes lies in their manipulation, especially in real-time. It often requires manipulating very large data sets, which results in a huge burden to current computing systems. Even though new techniques constantly emerge to allow better interaction with very large polygonal models, the problem of using the minimum number of polygons while meeting the maximum complexity requirement for a given representation precision still remains open. Moreover, on polygon meshes, it is difficult to directly maintain and evaluate surface continuity.

To alleviate the inherent difficulties of using traditional parametric B-reps and polygon meshes to represent geometry, we propose using a multi-resolution subdivision-based representation as the unifying technique to represent the underlying geometry and to support

adaptive mesh refinements and geometry modification for engineering analysis. We expect the integration of multi-resolution subdivision-based representations to link computational analysis, geometric modeling, refinement and visualization operations, which avoids introducing errors of higher degree and costing extra efforts to fill the incompatibility gap.

1.3 Advantages of Subdivision-based Representations

Subdivision-based representations, as a unifying geometric representation, merge the advantages of parametric B-reps and polygon meshes (Zorin, Holst et al. 1997, Zorin, Schroder et al. 1997, Zorin, Schroder et al. 2000). In fact, subdivision-based representations evolve from polygon meshes, and that is why the topological descriptions of subdivision-based representations are collections of ordered discrete geometric elements: vertices, edges and faces. Subdivision-based representations are generated by recursively applying subdivision operators on initial coarse polygonal meshes, which are defined as 'control meshes'. Thus they inherit directly the advantages of polygon meshes in representing and handling geometric models with arbitrary topology while achieving fine details.

Subdivision-based representations distinguish themselves from polygon meshes by the fact that the subdivision operators used for iteratively generating subdivision meshes are evolved from the subdivision masks based on B-splines or Box-splines (Barthe 2005, Dodgson, Augsdorfer et al. 2009). In practice, subdivision techniques can be viewed as refinement or discretization methods, since the new subdivided meshes are generated by discretizing the existing mesh elements, such as triangular faces in surface meshes or tetrahedrons in volume meshes. Consequently, new vertices are inserted between existing control vertices on control meshes through refinements, where the mesh density of the refined meshes is accordingly increased. The new inserted vertices greatly depend on the existing neighbouring vertices, and the relations between the new and existing vertices are defined by the subdivision masks above. This property insures that the refined meshes are conformal to their former meshes, which is essential for geometry-preserving refinements.

The subdivision techniques primarily serve to smooth meshes. In theory, after an infinite number of subdivision iterations, the subdivided meshes converge toward a smooth surface, usually named 'the limit'. The surface continuity of this limit is provable to be C^1 continuous or

C^2 continuous, depending on the applied subdivision scheme. This guaranteed continuity property of subdivision-based representations is desirable in mesh generation and mesh adaptation, since it is important to preserve surface continuity particularly when local parts or boundaries on meshes are being interactively modified.

In Stam's works (Stam 1998, Stam 1998), evaluation algorithms are developed to exactly and directly calculate the limit position of any arbitrary parameter value on Loop or Catmull-Clark subdivision surfaces without recursively applying the subdivision matrix. These subdivision surface evaluation algorithms also known as 'parameterizations on subdivision surfaces', allow obtaining analytic geometry information, such as surface coordinates, curvatures and derivatives, on subdivision-based representations (Persson, Aftosmis et al. 2006).

Some applications require sharp features on surface meshes to be preserved during mesh manipulation. In response to this requirement, special rules for creating creases, which can be viewed as boundaries between mesh patches were introduced (Zorin, Schroder et al. 2000, Biermann, Martin et al. 2002). The basic idea of creating creases is to keep sharp features of interior edges on subdivision surfaces, so that even after many subdivision iterations, the creases are not smoothed completely. In practical implementations (Wang 2006), the sharpness definition of creases can be flexibly controlled by adjusting the influence of neighbouring vertices around the edges defined as 'creases'.

In summary, subdivision-based representations combine the advantages of polygon meshes and parametric B-reps. This aspect underlines their unique capacity of representing complex smooth geometric models with arbitrary topology, supporting geometry-preservation refinements, maintaining surface continuity during shape modifications, supplying geometric information for engineering analysis, such as derivatives and curvature at arbitrary parameter values, and preserving sharp features according to user-specification.

1.4 Challenges of Utilizing Loop Subdivision-based Representations

This research is particularly aimed at limit surfaces generated by Loop subdivision. One reason is that Loop subdivision surface limits are C^2 continuous. The other reason is that the parameterization techniques on Loop subdivision surface limits are mature enough to be used as surrogates to CAD or analytic-based geometry (Persson, Aftosmis et al. 2006). Moreover,

triangular and tetrahedral meshes are commonly utilized for representing, simulating and visualizing surface meshes or volume meshes in geometric modeling. Our study indicates that Loop subdivision can be effectively used for refining these two types of meshes. Based on all these reasons, the subdivision representation in our research is exclusively chosen to be based on Loop subdivision.

During the integration of Loop subdivision evaluation techniques, it was found that computing subdivision limits is particularly difficult on boundary meshes when the evaluated triangular patches contain more than one extraordinary vertex or fixed vertex crease or edge crease definitions. This reality directs us to study the feasibility and efficiency of calculating the limit positions on irregular meshes through comparing the various existing subdivision surface evaluation methods.

Traditionally, Loop subdivision-based representations have emerged from repetitive subdivision sequences. In practice, this fact poses two disadvantages:

1. As subdivision levels increase, the numbers of vertices on the corresponding subdivision meshes expand exponentially. This could expose the computational system to the burden of storing redundant data.

2. The Level Of Details (LODs) on subdivided meshes is uniformly distributed, and this could be unfavorable for precisely representing local geometric features on selected mesh parts (Muller and Jaeschke 1998).

These two disadvantages are even more predominant in solid modeling. As we know, compared to surface meshes, tetrahedral meshes have a much more complex topology, and volumetric data are much more massive. Besides, recent researches show that solid subdivisions have not been well developed and subdivision techniques have not made an evident impact on engineering applications.

Moreover, the common issues related to using subdivision surfaces for engineering analysis (Gonsor and Neamtu 2001, Cirak, Scott et al. 2002, Ma 2005), such as surface parameterization, approximation accuracy, curvature representation, surface continuity and behaviour near extraordinary vertices, also motivate our research in integrating multi-resolution Loop subdivision-based representations for mesh refinement and geometry modification.

1.5 Hypotheses and Objectives

The general research objective is to develop a new geometric representation unifying geometry and mesh adaptive refinement for numerical simulation in engineering analysis purposes. This geometric representation is based on Loop subdivision. To support the efficient generation of multi-resolution Loop subdivision representations, a new refinement method: a single-level refinement method is developed. This general objective is fulfilled by assuming two conditions as below:

> C1: A multi-resolution subdivision mesh is used to surrogate the underlying geometry of a scanned model. Its boundary mesh is on a subdivision limit. Here one question arises: how to obtain the initial control mesh through retracing from this subdivision limit? To answer this question involves a large range of other researches, such as the reverse subdivision algorithms (Hassan and Dodgson 2005, Lanquetin and Neveu 2006, Sadeghi and Samavati 2011) or scattered data approximation techniques (Marinov and Kobbelt 2005). Since these researches are beyond the scope of the present study, it is assumed that the question above is pre-solved.
>
> C2: The refinement criteria used for guiding adaptive subdivisions in our work are pre-defined for demonstrating refinement results. The second condition is that these criteria can be defined by the users according to concrete engineering analysis requirements in real applications (Fidkowski and Darmofal 2011).

Under the two conditions above, the first hypothesis is specified as that the multi-resolution Loop subdivision-based representations obtained in our work are guaranteed not only to respect the geometry requirements but also to improve the representation accuracy. The second hypothesis is specified as the single-level refinement method can be used as a refinement method for precisely and efficiently obtaining the representation meeting users' specifications.

The specific objectives are:

1. Develop a solid subdivision scheme particularly for smoothing boundary surfaces;
2. Develop a subdivision limit projection algorithm for obtaining the analytic information of any arbitrary position on boundary meshes;

3. Assure maintaining adequate surface continuity on the newly adapted multi-resolution subdivision meshes;
4. Assure the meshes generated from adaptive subdivisions not to be degenerated in the context of numerical simulations;
5. Develop a refinement method for naturally integrating both adaptive refinements and subdivision techniques;
6. Assure that this developed refinement method can be intuitively applied on the other types of subdivision meshes.

1.6 Organization of the Thesis

This thesis is organized as follows. Chapter 2 presents a literature review covering the basic concepts of subdivision techniques, the essential properties of subdivision operators, the detailed Loop subdivision surface scheme and subdivision surface parameterization techniques and particularly Stam's exact evaluation method on Loop subdivision surfaces. In the second sub-section, we present several common geometric representations in solid modeling, address the advantages of solid subdivision techniques and study the recent approaches in solid subdivision schemes. In the third sub-section, we review the related works in multi-resolution subdivision-based modeling in three categories: mesh simplification, mesh refinement and LODs control. In the fourth sub-section, we investigate the potential of using multi-resolution subdivision surfaces for engineering purposes and present some recent related works. At the end of this chapter, we restate our research proposal for solving the current issues found in the actual implementations of solid subdivisions, subdivision surface parameterizations and adaptive refinements.

In Chapter 3, first we introduce the 1-to-8 topological split, the geometric smoothing based on Loop subdivision together with the rules of preserving sharp features (fixed vertex creases and edge creases) on boundary surfaces of tetrahedral meshes. Next we discuss the challenges of implementing this solid Loop subdivision in practice, particularly related to data structure and mesh quality evaluation, which demonstrate that uniform subdivision and approaching iterations are not evident in meeting precision and efficiency requirements. Adaptive subdivision is then proposed as a means to solve the problems above together with a discussion of the difficulties of integrating adaptive refinements in solid modeling.

In Chapter 4, the single-level refinement method is compared with the multi-level refinement method. This single-level method is proposed for the purpose of naturally merging the topological splits on whole subdivision-based meshes and the geometric smoothing on the corresponding boundaries. The implementation of the single-level refinement method is applied to curves, triangular surfaces and tetrahedral meshes.

In Chapter 5, two subdivision surface parameterization methods: Stam's exact evaluation method combined respectively with the mirroring technique and Persson's evaluation method are introduced for alleviating the limitations of the existing methods as discussed in Chapter 2. Then the refined mesh results obtained from applying both crease definitions and adaptive subdivisions on the representative examples are presented. Finally, the refinement results are analyzed and validated by evaluating mesh quality and time costs.

In Chapter 6, our contributions are summarized and the perspectives related to reverse subdivision, mesh quality evaluation and mesh connectivity calculation are discussed.

CHAPTER 2

STATE OF THE ART

To investigate the feasibility of utilizing the multi-resolution Loop subdivision limits as the fundamental geometric representation to unify modeling, meshing, analyzing, refining and visualizing in numerical simulations, we place our research in a larger context: subdivision techniques, subdivision techniques in solid modeling, multi-resolution techniques and control of mesh Level Of Details (LODs). In section 2.1, at first, we present the basic concepts of subdivision techniques and clarify the related terms: multi-resolution and LODs. Then we summarize the essential properties of the subdivision operators and state the advantages of implementing subdivision techniques to solve some of the relevant computational issues in geometric modeling. Next we present the Loop subdivision rules in details. At the end of this section, we study the subdivision limits deducted from the traditional subdivision iterations and introduce exact evaluation techniques on Loop subdivision surfaces. In section 2.2, we investigate the recent approaches of using subdivision techniques in solid modeling. In section 2.3, we present a survey of multi-resolution techniques related to subdivision surfaces, where the mesh LODs controlling technique is introduced. In section 2.4, we explore the potential capability of using multi-resolution subdivision techniques especially for engineering applications. At the end of this chapter, we restate the research proposal by addressing the problems in the related works.

2.1 Multi-resolution Subdivision-based Representations

The basic idea behind subdivision-based representations is to approach smooth representations by recursively applying subdivisions on coarse control meshes (Zorin, Schroder et al. 2000). The iterative subdivision process is an infinite refinement sequence, where the iteration levels are decided by the concrete requirements of the fineness and the smoothness on the subdivided meshes. The transmissions between the subdivision meshes at the sequential subdivision levels are performed by carrying out the subdivision operators S (see Equation 2.1):

$$M^{j+1} = S \cdot M^{j},\ (j \in N, M^{j} \subset R^{d}). \tag{2.1}$$

, where M^j represents the subdivision mesh at the j^{th} subdivision level. The definitions of the subdivision operators S depend on their corresponding subdivision rules, which formally are represented as subdivision matrices or subdivision masks.

2.1.1 Basic Definitions and Terms

The subdivision-based representation M^j generated by applying subdivision once, whose mesh density relates to its corresponding subdivision level j. In geometric modeling, the term 'resolution' is used to represent mesh density as determined by the number of control vertices in a subdivision mesh. Evidently, multiple subdivision meshes with various resolutions are generated during a subdivision sequence approaching the subdivision limit L (see Figure 2.1).

Figure 2.1: The illustration of the infinite subdivision sequence.

The sequence of these multiple subdivision meshes are refinement results of the same initial control mesh, each providing an approximation of the initial geometric form at the specified resolution. Thus, the subdivision techniques naturally support the generation of multiple subdivision meshes at various resolutions, where the Level Of Details (LODs) or density of each subdivision mesh is uniform. The multi-resolution subdivision-based representation proposed in this work is composed of several mesh parts, whose regional LODs are different or non-uniform.

Our subdivision-based refinement work involves one, two and three dimensional multi-resolution meshes: curves, triangular surface meshes and tetrahedral meshes. The general term "entity" is used to define the basic elements onto which a refinement is applied. There are three types of refinable entities: edges $E(v_i, v_j)$, triangular faces F_k^j and tetrahedrons T_k^j (see Figure 2.2).

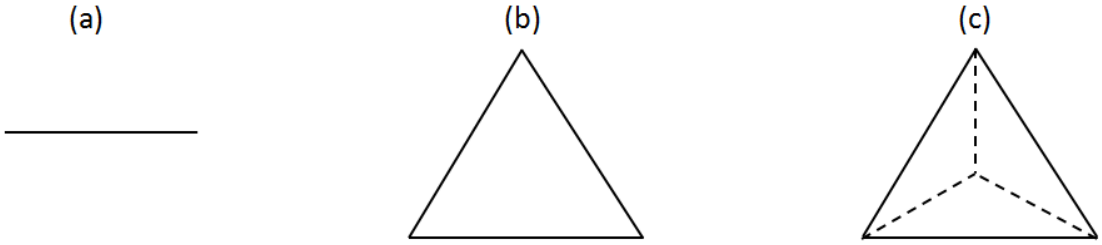

Figure 2.2: Three types of refinable entities: (a) edges, (b) triangles, (c) tetrahedrons.

In the context of subdivision-based refinements, a multi-resolution subdivision mesh can be viewed as the collection of refinable entities, whose LODs are determined by the numbers of refinable entities on local mesh parts.

For subsequent uses, we list the notations related to multi-resolution subdivision-based representations in Table 2-1.

Table 2-1: The list of notations related to subdivision-based representations.

Notation	Explanation
M^0	The initial coarse control mesh
M^j	The subdivision mesh at the j^{th} level, where $M^j \subset R^d$
S	The subdivision operator
L	The subdivision limit
v_k^j	The k^{th} control vertex at the j^{th} subdivision level, where $v_k^j \in R^d$
$s_{i,k}$	The i^{th} subdivision matrix coefficient applied on the k^{th} vertex
$E(v_i, v_j)$	The edge segment between two neighbouring vertices (v_i, v_j)
F_k^j	The k^{th} surface face at the j^{th} subdivision level, where $F_k^j \subset M^j$
T_k^j	The k^{th} tetrahedron at the j^{th} subdivision level, where $T_k^j \subset M^j$

2.1.2 Properties of Subdivision Operators

The subdivision operator S is a combination of two sub-functions: the topological split function $Split()$ and the geometrical smoothing function $Smoothing()$ (Warren and Weimer 2002). In actual operations, this combination can be specified as two sub-processes: 1) uniformly inserting new vertices (odd vertices) in-between the existing control vertices (even vertices); and 2) geometrically smoothing all the vertices by updating their positions. Figure 2.3 illustrates the detailed subdivision procedure on a square curve containing four control vertices.

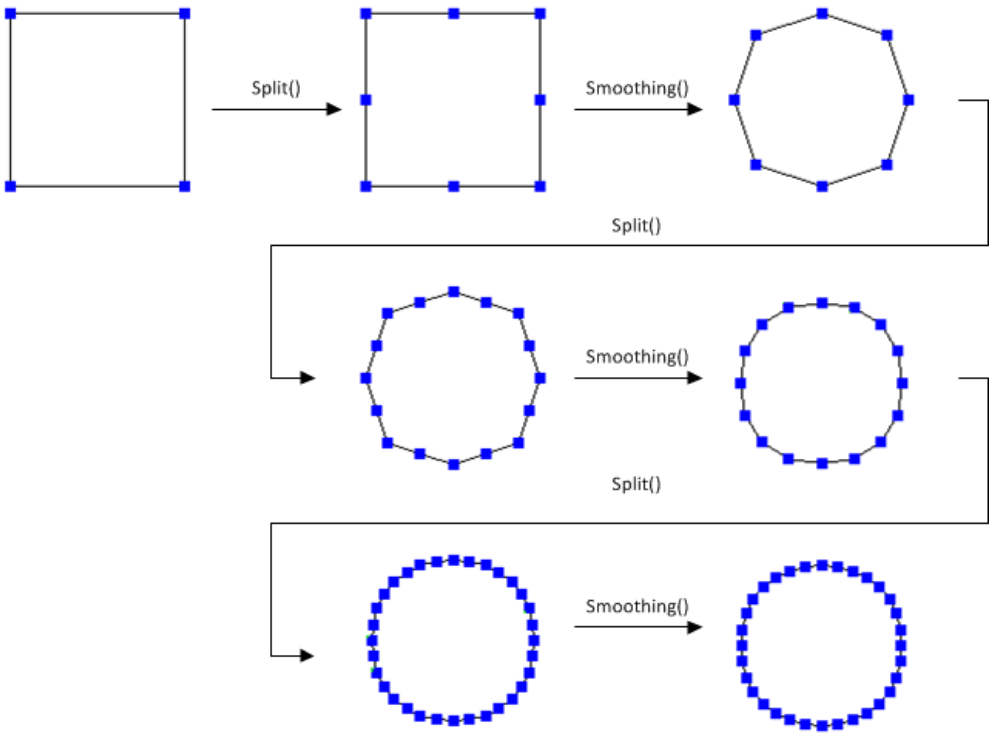

Figure 2.3: The subdivision operator S on a square curve as the combination of two sub-functions: split() and smoothing() (Schaefer , Warren and Weimer 2002).

From Figure 2.3, the two sub-processes above can be viewed as operations on vertices. Here, we need to emphasize that in the context of our refinement work, the primitive geometric elements on different dimensional multi-resolution subdivision meshes are always vertices. Thus, a multi-resolution subdivision mesh M^j at the j^{th} subdivision level can be defined as a sequence

of control vertices v_k^j (structured or unstructured), which span over the domain R^d (see Equation 2.2).

$$M^j = \left(v_k^j\right)_{0\le k<n} = \left(v_0^j, v_1^j \cdots v_k^j, v_{k+1}^j \cdots v_{n-1}^j\right),\ (j \in N). \tag{2.2}$$

In general, there are two types of subdivision schemes: approximation and interpolation. To distinguish between the above, we can verify if the geometric positions of the existing control vertices are changed during the geometric smoothing. The subdivision applied on the square curve example in Figure 2.3, is an approximation scheme. In order to clarify the difference between these two types of subdivision schemes, the results of applying an approximation subdivision scheme (Loop subdivision) and an interpolation subdivision scheme (Modified Butterfly subdivision) on the same example: a cone surface mesh are presented in Figure 2.4. The initial control mesh of the cone surface mesh is displayed in (a), while sub-figures (b-c) and (d-e) are respectively bottom view and side view. The subdivision meshes in (b.1) and (c.1) are subdivided once, in (b.2) and (c.2) are subdivided twice, and in (b.3, c.3, d, e) are subdivided three times. The results in the top row are subdivided using Loop subdivision, while the results in the bottom row are subdivided using Modified Butterfly subdivision.

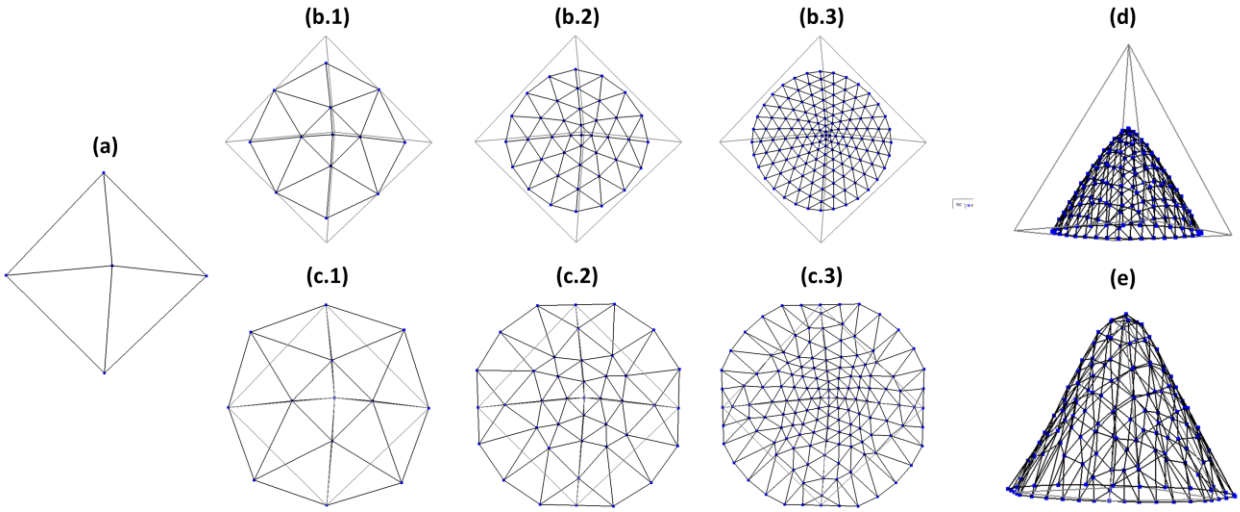

Figure 2.4: Comparison between an approximation subdivision scheme (Loop, top row) and an interpolation subdivision scheme (Modified Butterfly, bottom row).

This example shows that after iteratively applying an approximation subdivision scheme, the newly generated subdivision meshes are shrunk towards the interior of their original control meshes, whereas the iteratively interpolated subdivision meshes are inclined to keep the basic forms of their original control meshes.

Since both approximation and interpolation subdivision schemes involve the sub-function $Split()$, they can be classified as edge-based bisection schemes. In the traditional refinement process, repeatedly bisecting all the edges on control meshes results in the LODs of the new generated subdivision meshes being gradually and uniformly increased. That is to say, the number of control vertices on the subdivided mesh is multiplied after each subdivision. Therefore the control mesh M^j is a subset of the new subdivided mesh M^{j+1} (see Equation 2.3), where the existing control vertices are linearly mapped onto the next subdivision level.

$$M^j \subset M^{j+1},\ (j \in N). \tag{2.3}$$

The definitions of the discrete mappings between different subdivision levels are expressed in the subdivision matrices/masks during implementation. More specifically, the coefficients $s_{i,k}$ in the subdivision matrices build the relations between the control vertices at the j^{th} subdivision level and the control vertices at the $(j+1)^{th}$ subdivision level (Zorin, Schroder et al. 2000) (see Equation 2.4).

$$v_i^{j+1} = \sum_k s_{i,k} \cdot v_k^j,\ (i, j \in N). \tag{2.4}$$

Thus, in order to achieve the desired subdivision-based refinement results, it is important to study the properties of the subdivision operators. In the works of Zorin, Schroder et al. (Zorin, Schroder et al. 1997, Zorin, Schroder et al. 2000, Schroder 2004), the essential properties of the subdivision operators S are summarized as follows:

Compact Support. An initial control point has a relatively small influence on the shape of the final subdivision representations (see Equation 2.5).

$$\exists l_0, l_1 \in N \Rightarrow \forall k : s_{i,k} = 0, i \notin \{2k - l_0, 2k + l_1\},\ (i \in N). \tag{2.5}$$

Local Definition. The calculation of a new vertex's position only depends on the local

neighbors of its initial control vertex (see Equation 2.6). The relations presented in Equation 2.4 are further specified in Equation 2.6.

$$v_i^{j+1} = \sum_k s_{i-2k,k} \cdot v_k^j, \ (i, j \in N). \tag{2.6}$$

Affine Invariance. The transformations, such as translation, rotation and scaling on the initial control vertices are constantly preserved during subdivision (see Equation 2.7).

$$a \cdot v_i^{j+1} + \vec{b} = \sum_k s_{i,k} (a \cdot v_k^j + \vec{b}), \ (\vec{d} \in R^d, a \in R). \tag{2.7}$$

Index Symmetry. The subdivision matrices are symmetrically indexed (see Equation 2.8).

$$\begin{aligned} s_{i,k} &= s_{i+2,k+1}, \\ s_{2k-l,k} &= s_{2k+l,k}, \end{aligned} \quad (i,k,l \in N). \tag{2.8}$$

These four properties of the subdivision operators lay the foundation for subdivision methods to be competitive in addressing the important computational and topological issues to which geometric modeling practitioners are confronted. Firstly, the ‘compact support’ property is particularly favorable in achieving efficient adaptive subdivision-based refinements. The reason is that when some controls vertices on subdivision meshes are modified, only the information of the locally influenced vertices is updated instead of the whole mesh (Wang 2006). Secondly, the ‘local definition’ property supports rapid computation. The reason is that when new vertices are inserted on refined meshes, only their neighbouring vertices are required for obtaining the positions of the inserted ones. Thirdly, the ‘affine invariance’ property is essential in supporting a unifying geometric representation in the context of multi-resolution refinements. Fourthly, the ‘index symmetry’ property makes the calculation of subdivision matrices feasible and fast, which directly supports the efficient evaluation of the new generated subdivision meshes. In brief, all these properties are useful not only for analyzing subdivision matrices, but also for supporting the continuity and the convergence of subdivision limits (Zorin, Schroder et al. 2000, Wang 2006).

2.1.3 Loop Subdivision Scheme

In this work, the subdivision operator S is based on the Loop scheme, which was proposed by Charlie Loop around 1987 (Loop 1987). The subdivision rules of the Loop scheme are specified for two distinct triangular surface parts: 1) interior and 2) boundary or crease. For each surface

part, the subdivision masks of the new odd vertices and the existing even vertices are respectively defined. In a general term, on triangular subdivision surfaces, the interior vertices with valence 6 and the boundary vertices with valence 4 are regular. For example, on the cone subdivision surface mesh (see Figure 2.5), there are two interior vertices: V_0, V_1 and two boundary vertices: V_2, V_3 (see Figure 2.5(b)). Here, V_1 and V_3 are regular, while V_0 and V_2 are irregular.

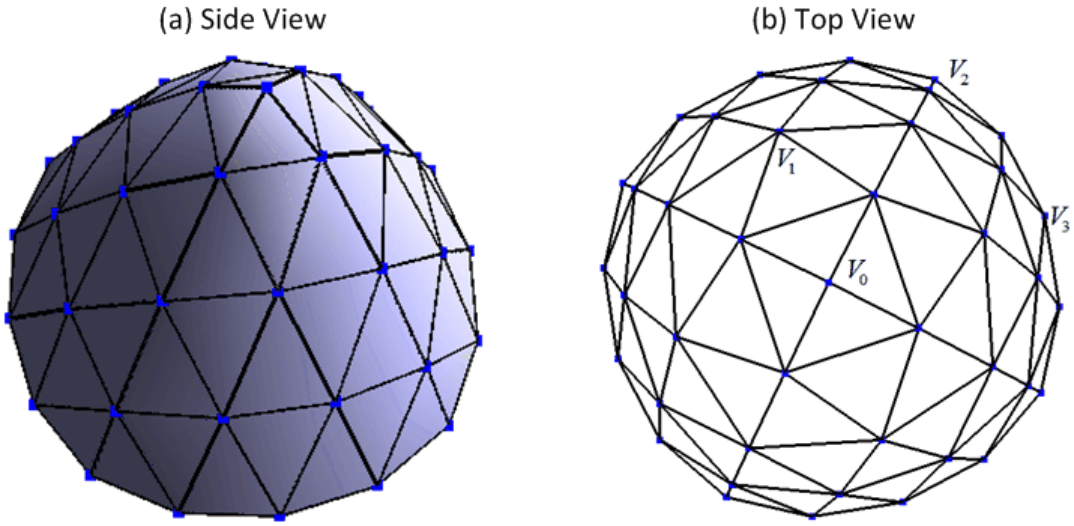

Figure 2.5: The cone subdivision surface mesh displayed in two modes: (a) shading mode with side view, (b) wire mode with top view.

Interior. The calculation of each odd vertex's position is based on 1) the two endpoints on the edge for split and 2) the other two vertices on the neighbouring triangles, which share the split edge. The geometric update of each interior even vertex is based on its neighbouring vertices and its own position at the lower subdivision level. The subdivision masks of the odd interior vertices and the even interior vertices are illustrated respectively in Figure 2.6(a) and Figure 2.6(b) (Zorin, Schroder et al. 2000).

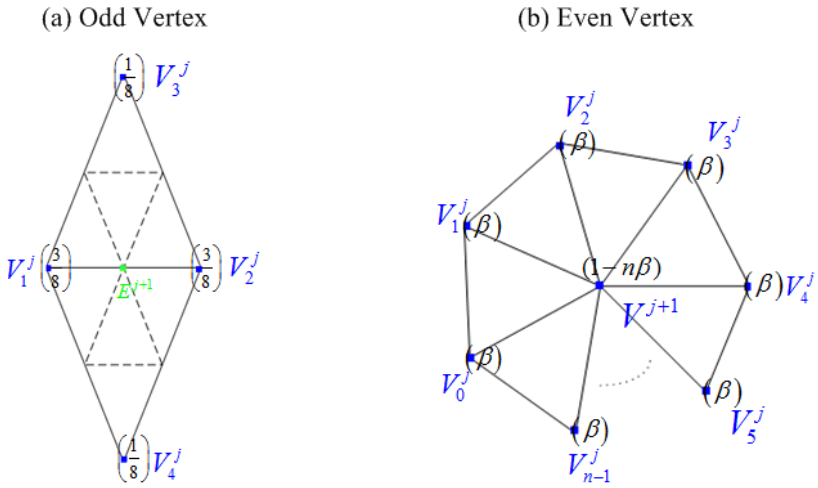

Figure 2.6: (a) the subdivision mask of the odd interior vertex E^{j+1}; (b) the subdivision mask of the even interior vertex V^{j+1} (Zorin, Schroder et al. 2000).

The concrete subdivision rule of each odd interior vertex E^{j+1} is described in Equation 2.9, where E^{j+1} represents the new inserted edge vertex at the $(j+1)^{th}$ subdivision level and V_i^j represents the i^{th} control vertex at the j^{th} subdivision level.

$$E^{j+1} = \frac{3}{8}(V_1^j + V_2^j) + \frac{1}{8}(V_3^j + V_4^j). \tag{2.9}$$

Similarly, the subdivision rule of each interior vertex V^{j+1} is described in Equation 2.10, where the coefficient β is used to define the weights of the neighbouring vertices V_i^j and its own position at the lower subdivision level V^j.

$$V^{j+1} = (1-n\beta)V^j + \beta\sum_{i=0}^{n-1} V_i^j \tag{2.10}$$

The definition of β influences the continuity and smoothness of subdivision surfaces, whose original version was proposed in (Loop 1987) (see Equation 2.11).

$$\beta = \frac{1}{n}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right). \tag{2.11}$$

However, using Loop's β definition could bring up one problem: when n equals 4 or 5, discontinuous but bounded curvatures emerge on subdivision surfaces. A simple corrected version of β was proposed to solve this problem (Warren and Weimer 2002) (see Equation 2.12). In this refinement work, this corrected version is used.

$$\beta = \begin{cases} \dfrac{3}{16} & (n = 3), \\ \dfrac{3}{8n} & (n > 3). \end{cases} \tag{2.12}$$

Boundary/Crease. The subdivision rules of the odd boundary vertices and the even boundary vertices are respectively illustrated in Figure 2.7(a) and Figure 2.7(b). These rules were firstly proposed by Hoppe et al. in (Hoppe, DeRose et al. 1994) for maintaining C^1 continuity on the subdivision surfaces with creases (Zorin, Schroder et al. 2000).

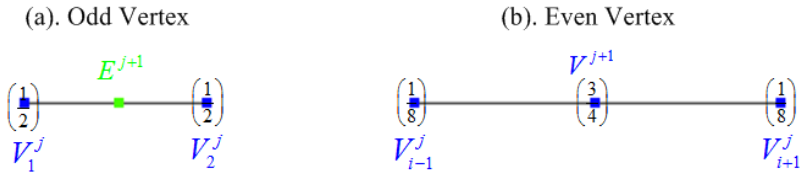

Figure 2.7: (a) the subdivision mask of the odd boundary vertex; (b) the subdivision mask of the even boundary vertex (Zorin, Schroder et al. 2000).

Each new odd vertex E^{j+1} on boundaries is defined only by the two endpoints of the selected edge (see Equation 2.13).

$$E^{j+1} = \frac{1}{2}V_1^j + \frac{1}{2}V_2^j. \tag{2.13}$$

In the same way, the new position of each even vertex V^{j+1} on boundaries is mainly influenced by its own position at the lower subdivision level (see Equation 2.14).

$$V^{j+1} = \frac{1}{8}V_{i-1}^j + \frac{3}{4}V_i^j + \frac{1}{8}V_{i+1}^j. \tag{2.14}$$

In this section, we presented the Loop subdivision masks of the interior and the boundaries on triangular surfaces. The Loop subdivision rules play a fundamental role in generating the unifying multi-resolution subdivision-based representation in our refinement works, not only for surface modeling, but also for solid subdivision.

2.1.4 Loop Subdivision Limits

In Loop's work, the quartic box-splines functions-based subdivision scheme was developed to generate a smooth subdivision surface from a coarse triangular control mesh with arbitrary topology through an infinite number of recursive subdivisions. The generated smooth surface is the subdivision limit L (see Equation 2.15).

$$L = \lim_{j \to \infty} M^j, \ (j \in N). \tag{2.15}$$

Normally, surface continuity of limit Loop subdivision surfaces are C^2 continuous, except near irregular mesh parts containing extraordinary vertices and boundaries/creases, which are only guaranteed with C^1 surface continuity (Zorin, Schroder et al. 2000). That is to say, the second derivatives of Loop subdivision surface limits are continuous almost everywhere. This continuity property constitutes an attractive factor for engineering analysis, particularly in Finite Element Analysis (FEA) (Burkhart, Hamann et al. 2010). In our subdivision-based refinements, we use this subdivision limit to represent the true geometry of the modelled physical object.

In surface modeling, the initial triangular control mesh M^0, when considered as a three-dimensional polygonal complex, consists of a set of triangular faces F. Tracking through the subdivision operator S, the Loop subdivision surface at any subdivision level M^j can be affinely evolved from its initial control mesh M^0 (see Equation 2.16).

$$M^j = S \cdot M^{j-1} = S \cdot SM^{j-2} = S \cdot S \cdot SM^{j-3} = \underbrace{S \cdots S}_{j-1} \cdot SM^1 = S^j \cdot M^0, \ (j \in N). \tag{2.16}$$

Then, Equation 2.15 can be rewritten as Equation 2.17:

$$L = M^\infty = S^\infty \cdot M^0 = \lim_{j \to \infty} S^j \cdot M^0, \ (j \in N). \tag{2.17}$$

Equation 2.17 reveals two facts: 1) Loop subdivision surface limits can be obtained by repeatedly applying the Loop subdivision operators on their initial control meshes; 2) the LODs on the limits grow exponentially growing. In real applications, the first fact leads to redundant computations and the second fact could pose heavy burden on system resources.

The parameterization techniques on subdivision surfaces (Stam 1998, Stam 1998) establish a direct mapping between an initial coarse control mesh M^0 and its smooth subdivision limit L, where an infinite times of multiplying subdivision operators S^∞ is mathematically simplified through evaluation masks. That is to say, limit subdivision surfaces can be directly obtained by simply applying the evaluation masks on initial control meshes. In our refinement work, we denote the Loop subdivision surface parameterization by the projection function $f_p()$ (see Equation 2.18).

$$f_p() : M^0 \to L. \tag{2.18}$$

Here, we illustrate applying the projection function $f_p()$ on a cube surface example (see Figure 2.8). Its initial control mesh M^0 consists of 12 triangular faces and its limit subdivision surface L consists of 12 triangle faces (see Figure 2.8(b) and Figure 2.10).

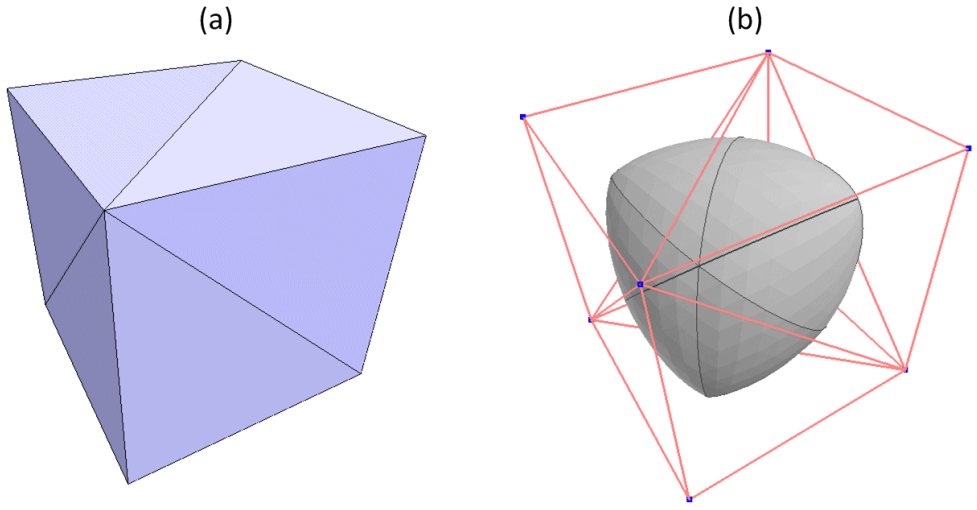

Figure 2.8: (a) The initial control mesh in shade mode, (b) The limit subdivision surface (in grey color) contoured by its initial control mesh (red wires)).

In a parameterized way, the topology of a limit subdivision surface L can be viewed as a union of triangular faces, where each triangular face is respectively associated with a parametric surface patch. The parametric form of each triangular patch is defined with parameters (v, w) over a computational parametric domain Ω, which Jos Stam specified as the "unit triangle" (Stam 1998) (see Equation 2.19).

$$\Omega = \left\{ (v, w) \mid v \in [0,1] \text{ and } w \in [0, 1-v] \right\}. \tag{2.19}$$

The unit triangle is formed by a barycentric coordinate system, where $v = 1$ corresponds to the point $(1, 0)$, $w = 1$ corresponds to the point $(0,1)$, $u = 1$ corresponds to the origin $(0, 0)$ and $u = 1 - v - w$ (see Figure 2.9(a)).

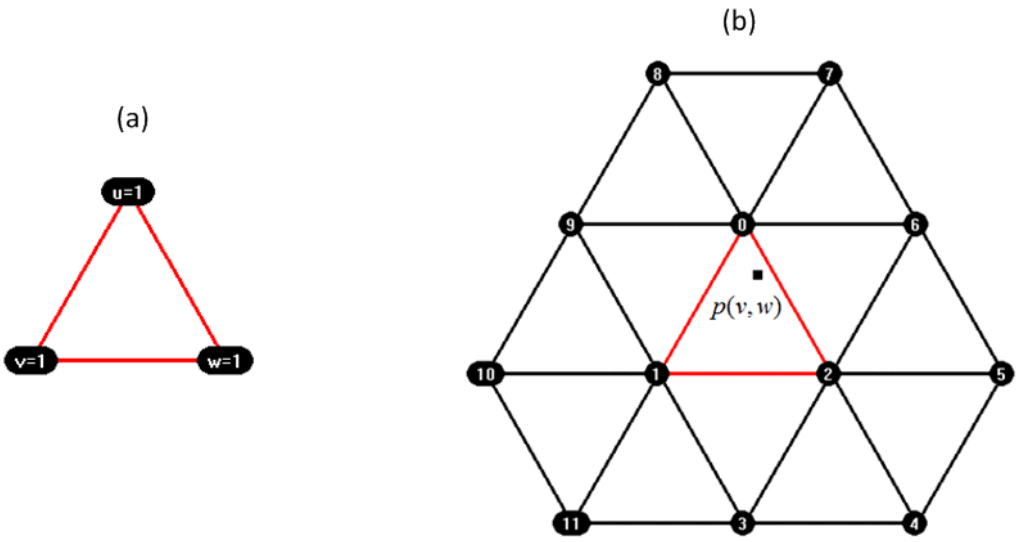

Figure 2.9: (a). The unit triangle defined in a barycentric coordinate system, where u, v or w = 1, (b). A regular triangular patch with 13 triangles and defined by 12 control vertices.

Any arbitrary location on a triangular face is indexed through two parameter values v and w in $p(v, w)$. To calculate its limit position means exactly evaluating the corresponding location on the limit subdivision patch. Figure 2.10 illustrates several exact evaluations of the above cube surface example using 12 triangle patches.

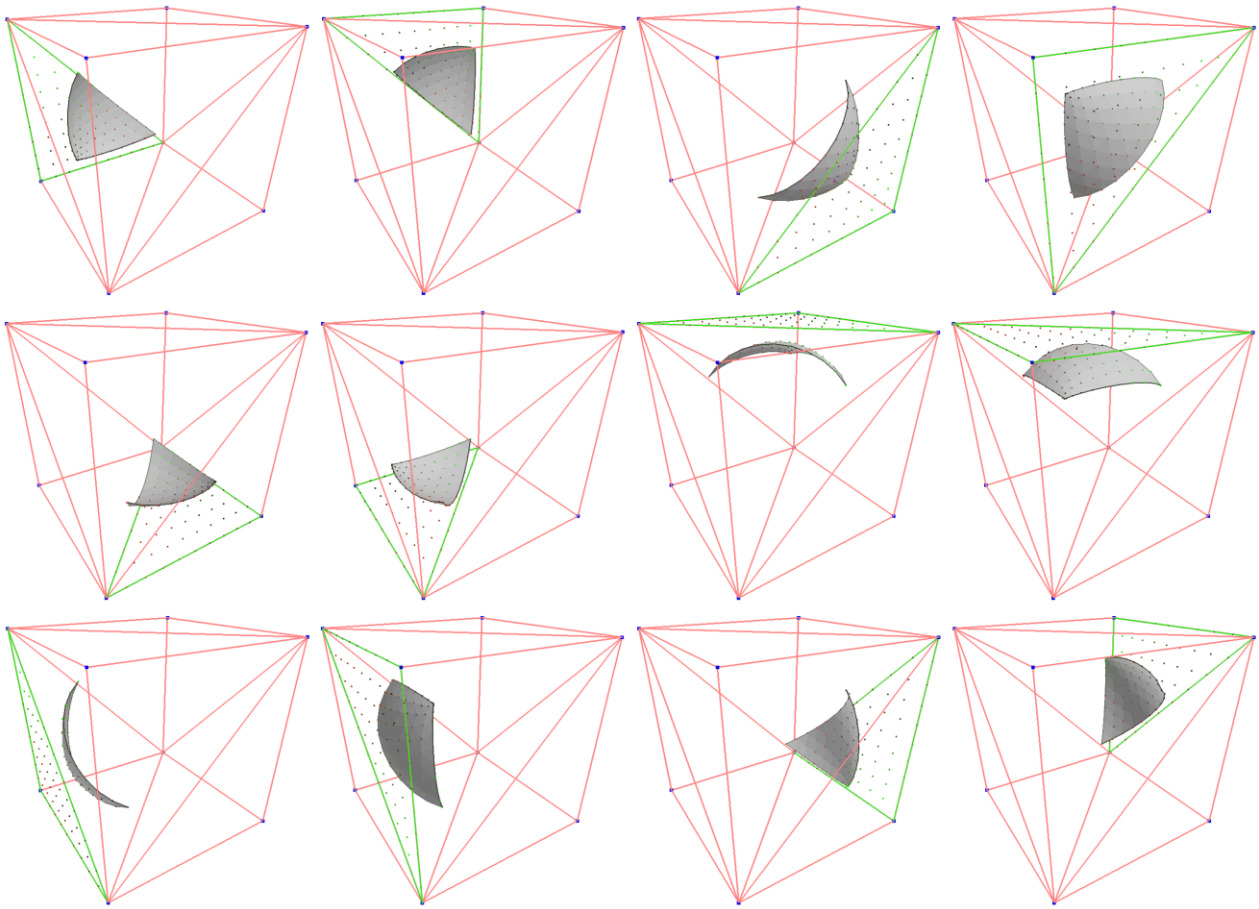

Figure 2.10: Exact evaluation of arbitrary locations on the generated limit cube subdivision surface. Each limit triangle face (displayed in grey shading mode) respectively corresponds to a triangular face in the initial coarse control mesh (displayed in green lines).

The difficulty level of parameterizing subdivision limits depends on the topological complexity of the evaluated local surfaces. Around 1992, Lai developed the B-nets matrix calculation algorithm (Ming-Jun 1992), which permits to directly parameterize topologically regular triangular meshes using triangular Bézier patches derived from a collection of convoluted bivariate box-splines.

Around 1998, Jos Stam in his work (Stam 1998) patternized a topologically regular triangular mesh as a collection of regular triangular patches. On triangular meshes, a regular vertex is directly connected to 6 neighbouring vertices. A single regular triangular face F contains 3

regular control vertices V_0, V_1, V_2 . and this face is in the immediate vicinity of 12 other triangular faces (see Figure 2.9 (b)). Thus, a regular triangular patch is defined by 12 control vertices.

In Jos Stam's work, this patch, based on 12 control vertices, is expressed in an evaluation matrix formed with 12 basis functions of degree 4 (see Equation 2.20). This expression is evolved from the B-net computing algorithm in Lai's work.

$$\frac{1}{12}\begin{bmatrix} 6 & 0 & 0 & -12 & -12 & -12 & 8 & 12 & 12 & 8 & -1 & -2 & 0 & -2 & -1 \\ 1 & 4 & 2 & 6 & 6 & 0 & -4 & -6 & -12 & -4 & -1 & -2 & 0 & 4 & 2 \\ 1 & 2 & 4 & 0 & 6 & 6 & -4 & -12 & -6 & -4 & 2 & 4 & 0 & -2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 6 & 6 & 2 & -1 & -2 & 0 & -2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & -2 & -1 \\ 1 & -2 & 2 & 0 & -6 & 0 & 2 & 6 & 0 & -4 & -1 & -2 & 0 & 4 & 2 \\ 1 & -4 & -2 & 6 & 6 & 0 & -4 & -6 & 0 & 2 & 1 & 2 & 0 & -2 & -1 \\ 1 & -2 & -4 & 0 & 6 & 6 & 2 & 0 & -6 & -4 & -1 & -2 & 0 & 2 & 1 \\ 1 & 2 & -2 & 0 & -6 & 0 & -4 & 0 & 6 & 2 & 2 & 4 & 0 & -2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 \end{bmatrix}_{12\times 15} \begin{bmatrix} 1 \\ v \\ w \\ v^2 \\ vw \\ w^2 \\ v^3 \\ v^2w \\ vw^2 \\ w^3 \\ v^4 \\ v^3w \\ v^2w^2 \\ vw^3 \\ w^4 \end{bmatrix}_{15\times 1} \tag{2.20}$$

In practice, this evaluation matrix can be directly utilized for calculating the limit position of any arbitrary location on regular triangular meshes. The major contribution of Jos Stam was to develop the evaluation algorithm which permits the parameterization on triangular subdivision surface meshes with irregular topology, such as a triangular face containing one extraordinary vertex (see Figure 2.11(a)).

The main advantages of Stam's Loop subdivision surface evaluation algorithm can be summarized as follows:

1. Usually, a limit position is computed through affinely multiplying the position of the evaluated vertex together with its neighbouring vertices on the initial control mesh by the

evaluation matrix (see Equation 2.17). In fact, the limit can be obtained as well from applying the same evaluation matrix on the refined mesh at any subdivision level, which is subdivided once or many times using the standard Loop subdivision rules (see Equation 2.21). Stam's evaluation algorithm addresses this fact well.

$$L = S^{\infty} \cdot M^{0} = \lim_{i \to \infty} S^{i} \cdot M^{j} \quad (i, j \in N). \tag{2.21}$$

2. Regarding the irregular patches on triangular meshes, Jos Stam isolates them by applying more subdivisions. After applying the standard Loop subdivision once, the patch containing one extraordinary vertex is partitioned into 4 sub-patches (see Figure 2.11(b)). Then from these partitioned sub-patches, Jos Stam summarizes that ¾ of these patches can be calculated according to the standard regular pattern (see Figure 2.12). There is still one irregular sub-patch left, which can't be matched with the regular pattern.

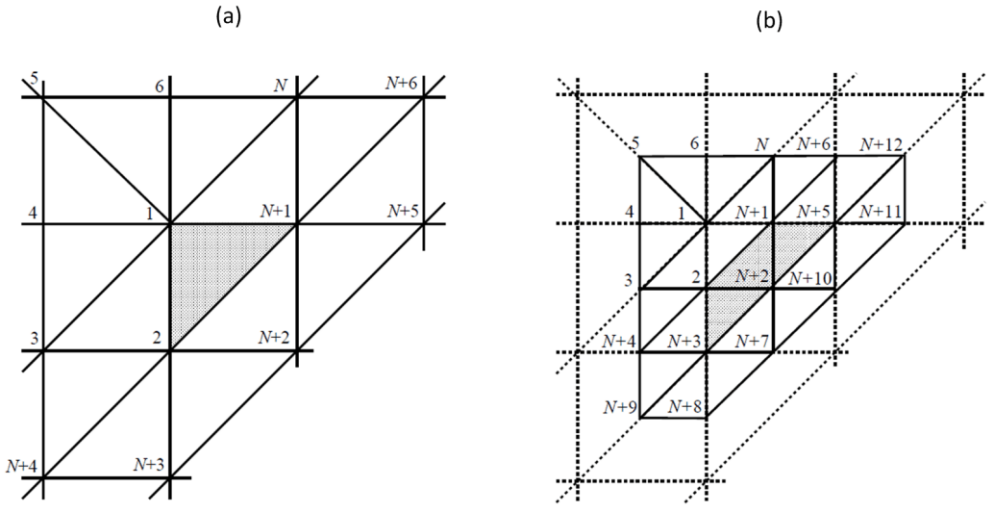

Figure 2.11: (a) One pattern of an irregular patch, which is defined by 13 control vertices. The vertex marked as '1' is an extraordinary vertex with valence != 6. (b) Applying once the traditional Loop subdivision, the original patch is partitioned into four sub-patches (Stam 1998).

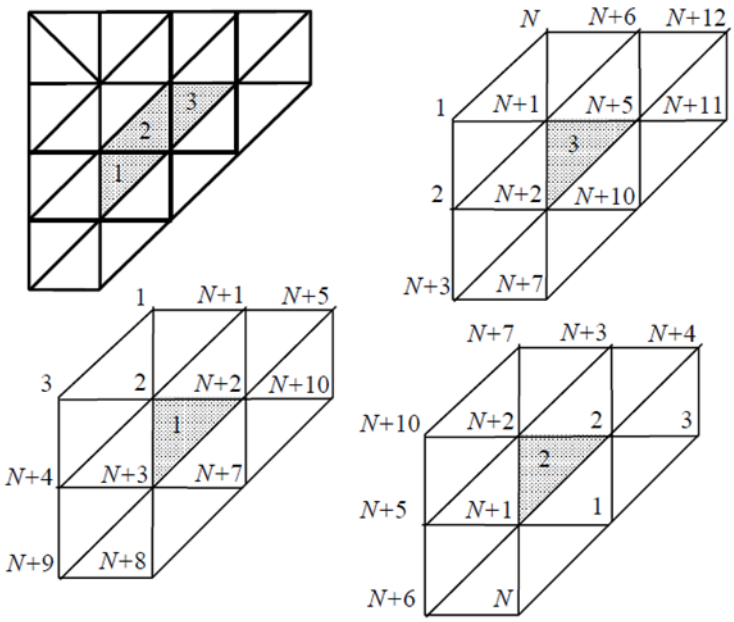

Figure 2.12: 3/4 of the partitioned sub-patches can be calculated using the regular pattern (Stam 1998).

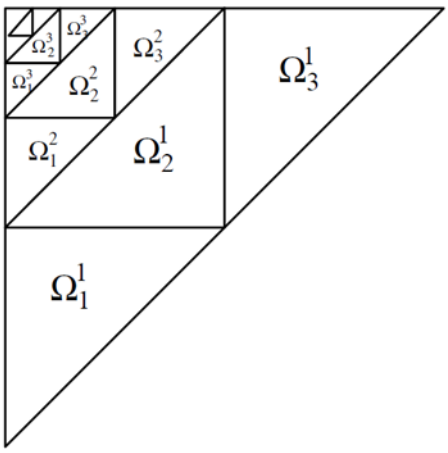

Figure 2.13: One patch is tiled into an infinite set of triangular tiles, where each tile is defined as one sub-domain Ω_1^i, Ω_2^i or Ω_3^i (Stam 1998).

3. Regarding the irregular sub-patch left, Stam applies the same isolation logic, "tiling" n times to find the result of any values of v, w (see Figure 2.13). In this tiling way, the irregular vertex is isolated into a smaller and smaller computation domain until it can be

calculated. Since this tiling partition is iterative, it yields a complex function that is expensive to calculate and may cause numerical instability. Instead, Stam decomposes this function into eigenvalues and vectors. For extraordinary vertices with valence from 3 to 50, Jos Stam pre-fills their corresponding eigenvalues and vectors in a binary file.

4. Finally, to evaluate the limit position of any point on a triangle of the control mesh involving the above eigenvalues, the triangle's vertices need to be projected into the eigenspace. This projection work can be done once when the mesh is initialized. In practice, Stam's evaluation function can be directly implemented into other systems, since Stam makes all the relevant source codes publicly available.

However, Stam's evaluation algorithm proposed in (Stam 1998) can be used to calculate the limit position in a triangular patch, with only one extraordinary interior vertex. For the triangular patches containing boundary vertices or with more than one extraordinary interior vertex, evaluation issues arise.

2.2 Solid Subdivision

The basic concept of subdivision schemes comes from Chaikin's corner cutting algorithm for generating smooth arbitrary curves from a limited number of control points (Chaikin 1974). It is cubic Catmull-Clark subdivision (Catmull and Clark 1978) and quadratic Doo-Sabin subdivision (Doo and Sabin 1978) which extended the development of subdivision schemes from curves to surfaces. Around 1998, for the first time subdivision surfaces were successfully utilized in modeling the animation characters of the short film 'Geri's Game' (Kerlow 2004). Since then, subdivision-based representations have gained their popularity in representing smooth surfaces with arbitrary topologies. However, until now the application of subdivision techniques in solid modeling has not been addressed much.

2.2.1 Current Geometric Representations in Solid Modeling

In solid modeling, a geometric object is defined by three parts: interior, exterior and boundary, and each part holds its own properties and features. Because of the topology complexity and the massive geometric data, it is much more challenging to represent three-dimensional solids in comparison to surface modeling. There are three common methods to represent geometric solids:

Constructive Solid Geometry (CSG), Boundary Representation (B-rep) and Cell Decomposition (Rossignac and Requicha 1999). Each solid geometric representation owns its proper data structure, mathematical operations and computation algorithms.

The CSG method represents complex solids as a combination of primitive sets, which are simple shapes, such as spheres, cubes, cones and pyramids. Then the relations of the primitive sets are built by using *Boolean* operators: unions ($\bigcup$), intersections ($\bigcap$) and differences (-). In order to describe complex functional primitives, such as distinguishing the interior solid from its boundary and the exterior, it is necessary to combine the CSG method with implicit functions. In this combined technique, one particular set of vertices, such as the set of boundary vertices, is implicitly specified by using predicates (see Equation 2.22).

$$f(x, y, z) = 0. \tag{2.22}$$

The combined technique named as Level Set Method (LSM) presents a major difficulty when there are complex inconstant boundaries on the simulated geometric solids. The numerical solutions become intractable if only resorting to the implicitly defined geometry information. Besides, in practice, this method is not suitable for direct manipulation and evaluation on the corresponding geometric representation.

Different from the CSG method, Boundary Representation (B-Rep) represents solid objects as collections of boundaries or limits by using parametric entities. Since solid B-Reps are constructed from a few control parameters, it allows flexible manipulation and effective evaluation on parametric solid representations. The representative B-Rep solid works include the Bernstein-Bézier representation of volumes (Lasser 1985), the trivariate tensor product B-Spline solids (Sederberg and Parry 1986, Griessmair and Purgathofer 1989, Joy 1991, Joy and Duchaineau 1999), and NURBS solid models (Gursoy 1996, Zhang, Bazilevs et al. 2007). Bézier, B-Splines and NURBS, these three parametric shapes are frequently used to represent smooth free-forms in numerical analysis. Essentially they share many similarities, while B-Spline representations are generalizations of Bézier representations and NURBS representations are further generalizations of B-Spline representations (Piegl and Tiller 1997).

The trivariate B-spline solid representation can be formulated as Equation 2.23 (Joy and Duchaineau 1999, Martin, Cohen et al. 2008):

$$V(u,v,w)=\sum_{i_0=0}^{n_0}\sum_{i_1=0}^{n_1}\sum_{i_2=0}^{n_2}P_{i_0,i_1,i_2}\cdot N_{i_0,m_0}(u)N_{i_1,m_1}(v)N_{i_2,m_2}(w), \tag{2.23}$$

where $\left\{P_{i_0,i_1,i_2}:\ 0\le i_0\le n_0,\ 0\le i_1\le n_1, 0\le i_2\le n_2\right\}$ are the control vertices on the $(i_0+1)\times(i_1+1)\times(i_2+1)$ control mesh, N_{i_j,m_j} are the normalized B-Splines functions of degree m_j defined on three sets of knot vectors $\{U,V,W\}$ (see Equation 2.24) and the three parametric variables $u\in\left[u_{m_0-1},u_{n_0+1}\right], v\in\left[v_{m_1-1},u_{n_1+1}\right]$ and $w\in\left[w_{m_2-1},u_{n_2+1}\right]$.

$$\begin{aligned} U&=\left\{u_0,u_1,\cdots u_{n_0+m_0}\right\};\\ V&=\left\{v_0,v_1,\cdots v_{n_1+m_1}\right\};\\ W&=\left\{w_0,w_1,\cdots w_{n_2+m_2}\right\}. \end{aligned} \tag{2.24}$$

From the equations above, the blending basis functions N_{i_j,m_j} are based on tensor-product. Tensor-product based schemes limit the corresponding parametric representations to utilize rectangular structures. This fact makes constructing smooth solids with arbitrary topology quite challenging. There are two particular reasons: 1) high degrees are required to keep surface continuity between different solid parametric patches; 2) expensive algorithms are required to construct various solid forms.

By using a cell decomposition method, a solid object is represented as the composition of spatial cells, which can be partitioned recursively. Octree is one representative cell decomposition method in solid modeling (Ito, Shih et al. 2009). In the Octree method, each solid can be subdivided into eight sub-solids, which are named as internal nodes. Then analogically, each sub-solid can be subdivided into eight child nodes. All these nodes are connected in a hierarchical tree. The tree structure evolved from the recursive decomposition can be used for keeping the hierarchical relationships between the existing nodes and the newly generated nodes. Eventually, this aspect is favorable for representing complex 3D geometry (Junye, Xiaoxian et al. 2008).

In our research, we propose using subdivision solids as the solid geometric representations. A subdivision solid can be defined as the limit of a sequence of successive refinements of

volumetric meshes. More specifically, a subdivision tetrahedral solid is constructed from four basic elements: vertex, edge, face and tetrahedron. The topological information of a subdivision solid is the connections among all these basic elements. Correspondingly, the iterative subdivisions on subdivision solids not only topologically update their mesh connectivity, but also geometrically smooth their boundaries. Similarly to subdivision surfaces, the solid subdivision operations define the parametric relations between the initial control vertices and the control vertices at the higher subdivision levels on solid boundaries. Thus, subdivision solids merge advantageous aspects of representations generated from B-Reps and Octrees.

2.2.2 Solid Subdivision

Until now, most works related to subdivision techniques have been developed for modeling surface meshes. The first solid subdivision work was presented in the work of MacCracken and Joy (MacCracken and Joy 1996). Their work used a tensor-product extension of Catmull-Clark subdivision to successively refine 3D hexahedral lattices. This Catmull-Clark subdivision-based refinement method innovatively brought the attention of deforming solid objects with arbitrary topology by implementing subdivision techniques.

It is the works of Bajaj et al. (Bajaj, Schaefer et al. 2002) and Chang et al. (Chang, McDonnell et al. 2002) that opened up new prospects of developing solid subdivision schemes for modeling free-form solids. Chang et al. in (Chang, McDonnell et al. 2002) presented an approximating solid subdivision scheme using trivariate Box-Splines to subdivide 3D tetrahedral meshes. However, the complexity of these trivariate Box-Spline rules impedes its utilization in approximating smooth tetrahedral meshes. Besides, this work involves generating octahedrons inside regular tetrahedral meshes, which limits its implementation on volume meshes with complex topology.

Around 2003, Chang et al. in (Chang, McDonnell et al. 2003) proposed an interpolating solid subdivision scheme, which uses a combination of simple linear interpolations to generate limit quadrilateral meshes. The surface continuity of the interpolated limits is proved to be C^1. All these early solid subdivision works have been proposed as methods of generating three-dimensional smooth deformable regions, but they are not accommodable to adaptive discretization.

Recently, Burkhart et al. (Burkhart, Hamann et al. 2010) have presented an approximating $\sqrt{3}$ subdivision scheme, which allows adaptively refining tetrahedral meshes. Their solid subdivision scheme implicates 1-to-4 split, face and edge flips, and $\sqrt{3}$ subdivision scheme-based geometric smoothing. The adaptation in this work aims for optimizing the mesh quality. It is Leif Kobbelt who developed the $\sqrt{3}$ subdivision scheme (Kobbelt 2000). Similarly to the Loop subdivision scheme, $\sqrt{3}$ subdivision works with triangular surface meshes and the surface continuities of their subdivision limits are nearly C^2 everywhere. However, the subdivision results in Burkhart et al.'s work are obtained by simply combining the $\sqrt{3}$ subdivision scheme and adaptive refinement criteria without iteratively approaching subdivision limits. This fact makes their geometric smoothing closer to approximation functions than the actual subdivision smoothing. Since in real applications, interactively adapting complex 3D tetrahedral meshes relates to two important issues: mesh quality and efficiency, the work of Burkhart et al. could be more comprehensive if its adapting efficiency was discussed.

From above, subdivision solids are in fact collections of volumetric subdivision entities. The computational complexity of solid subdivision mesh particularly depends on the number of volumetric subdivision entities. That is to say, as more volumetric subdivision entities are generated, the corresponding mesh resolution is increased at the cost of memory and time. So an optimal choice is to model subdivision solids with the minimum number of volumetric subdivision entities while meeting the maximum complexity requirements. As we know, traditional solid subdivision iteratively generates multiple solid subdivision meshes at different subdivision levels, where at each subdivision level, volumetric subdivision entities are uniformly distributed. To optimally use subdivision solids for modeling, it is essential to introduce adaptive subdivisions.

The implementation of adaptive subdivisions involves the generation of multi-resolution meshes, which directly implies works about mesh simplification, mesh refinement and LODs control. In the following section, such works related to multi-resolution subdivision-based modeling are going to be discussed.

2.3 Multi-resolution Subdivision-based Modeling

The basic concept of multi-resolution modeling begins with the introduction of hierarchical geometric models for visible surface algorithms (Clark 1976). This was an attempt to utilize hierarchical models in a general framework for providing various amounts of details in a scene with coherent frames.

Most of the current multi-resolution modeling works are about modeling polygonal surface meshes, which can be classified into two categories: mesh simplification and mesh refinement (see Figure 2.14).

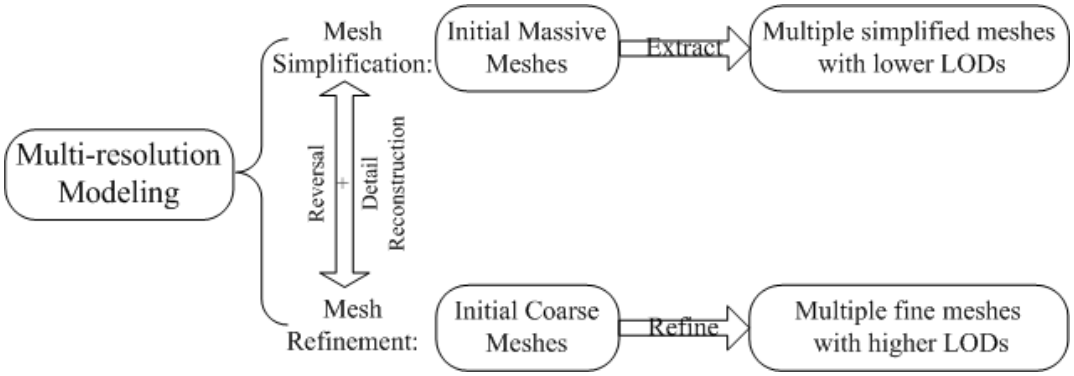

Figure 2.14: Multi-resolution modeling works: mesh refinement and mesh simplification.

2.3.1 Mesh Simplification

Mesh simplification is to approximately extract multiple simplified meshes with fewer details from an initial massive mesh model using simplification algorithms. Mesh simplification can be viewed as a constrained optimization process through the implementation of global or local simplification operators, where fidelity or triangle-budget is used as a constraint (Luebke, Reddy et al. 2002).

In order to obtain desirable results, it is important to measure approximation errors between the initial meshes and the final meshes during and after the simplification process. The error measurement methods definitively influence the quality of reconstructed meshes. A comprehensive comparison of mesh simplification algorithms is presented in the work of

Cignoni et al. (Cignoni, Montani et al. 1998). This comparison not only theoretically characterizes the fundamental mesh simplification methods but also empirically analyzes the computation complexities and the approximation accuracy of the experimental results in the different mesh simplification approaches. This work is a valuable reference for assessing the quality of approximated meshes.

The related approaches on simplification error metrics (Luebke, Reddy et al. 2002) include local geometric errors such as the calculation of the vertex-plane distance with the representative example: Quadric Error Metric (QEM) (Garland and Zhou 2005), the surface-surface distance (to calculate maximum, mean and mean square errors between two surfaces) using Hausdorff distance (Aspert, Santa-Cruz et al. 2002), and the vertex-surface distance (ex. Attribute Deviation Metric (ADM) (Roy, Nicolier et al. 2002)), and global error measures with both numerical and visual results (Cignoni, Rocchini et al. 1998). In addition, some software programs for measuring the distortion between two surface meshes, such as Measuring Error between Surfaces using the Hausdorff distance (M.E.S.H.) (Aspert, Santa-Cruz et al. 2002), QSlim (Garland 1999), Metro (Cignoni, Rocchini et al. 1998) and MeshDev (Roy, Foufou et al. 2006) are publicly available for evaluating surface mesh quality.

Michael Garland comprehensively surveyed the notable simplification algorithms on curves and surfaces in the context of multi-resolution modeling (Garland 1999). The polygonal surface simplification methods such as vertex clustering, region merging, wavelet decomposition, vertex decimations and iterative contraction, are discussed in details. In this report, the concept of evaluating surface approximation together with approximation criteria in rendering and finite element analysis applications is introduced.

In real applications, mesh simplification works involve many different issues in geometric modeling, such as quality/system performance, topological complexity, visual fidelity, geometric fidelity, the ability of preserving sharp features and LODs control. With respect to the geometric fidelity of the simplified mesh results, Surazhsky and Gotsman in their technical report (Surazhsky and Gotsman 2005) qualitatively compare some current mesh simplification software packages: 3DS Max, Geomagic Studio, Rapid Form 2004, Rational Reducer, Maya, Qslim, VizUp, Direct3D and Memoryless. Their geometric distortion comparison is guided by common geometric distance measures: Hausdorff distance and average distance. The test dataset in this

study, which includes several representative models with different sizes and properties, can serve as a useful reference in analyzing mesh results.

2.3.2 Subdivision-based Mesh Refinement

Mesh refinement aims to increase a mesh resolution and introduce more details into an initial coarse mesh using refinement algorithms. Subdivision surfaces as an efficient refinement method combined in the multi-resolution structures are quite useful for representing massive meshes with complex arbitrary topology.

The concept of Multi-Resolution Analysis (MRA) was introduced by Mallat et Meyer (Mallat 1989, Meyer 1993) and the theoretical basis of applying MRA to surfaces with arbitrary topology is established in Lounsbery et al's work (Lounsbery, DeRose et al. 1997). They formulated MRA by constructing two basic functions: a scaling function $\Phi_i^j(x)$ and a wavelet function $\psi_k^j(x)$, whose inner product equals zero (see Equation 2.25):

$$\left\langle \Phi_i^j(x),\ \psi_k^j(x) \right\rangle = \int \Phi_i^j(x)\ \cdot\ \psi_k^j(x) = 0. \tag{2.25}$$

In Equation 2.25, the scaling function $\Phi_i^j(x)$ is spanned in an infinite chain of hierarchical nested linear function spaces V^j where $V^0 \subset V^1 \subset \cdots \subset V^{j-1} \subset V^j \subset \cdots V^\infty$ and the wavelet function $\psi_k^j(x)$ as a translating function is orthogonal to the refinable scaling function $\Phi_i^j(x)$.

The work of Lounsbery et al. (Lounsbery, DeRose et al. 1997) detailed the close relation between subdivision surfaces and the scaling function and the wavelet function. This relation indicates that during subdivision iterations, the scaling function is replaced with the topological splits and the wavelet function is replaced with the geometrical smoothing. Thus the essentials of constructing two orthogonal functions can be directly extended as continuously subdividing coarse meshes and approximately controlling Level-Of-Details (LODs). This conclusion reveals that subdivision techniques naturally support the generation and the manipulation of large scale meshes with complex geometry.

In MRSS modeling, the early approaches mostly concentrate on interactive subdivision surface editing, where Peter Schroder and Denis Zorin make significant contributions. Around 1997, an interactive editing system based on subdivision surfaces in a multi-resolution

framework is presented by Zorin et al. (Zorin, Schroder et al. 1997). This editing system combines the analysis, synthesis and rendering algorithms with the local and adaptive operators, which allows to interactively edit large scale meshes with flexible adjustments of local details. Their results demonstrate that adequate interactivity is achieved. However, extra time for analyzing or extra memory for preserving its hierarchical structure is needed in this approach.

An umbrella algorithm combining local frame coding and multi-level smoothing, which permits manipulating arbitrary triangle meshes interactively, is presented by Kobbelt et al. in (Kobbelt, Campagna et al. 1998). Their major contribution is to develop an incremental mesh decimation schemes to generate hierarchical meshes without consulting subdivision connectivity. In this approach, mesh modification is constrained to discrete energy minimization functions and the influence of the editing region is predefined in the specified borders.

In Lee's work (Seungyong 1999), an editing process based on harmonic maps (mapping editing areas into 2D rectangles) and multi-level B-spline interpolation (updating the modification in 2D back to 3D meshes) is presented. The limitation of this innovative method comes from the calculation of harmonic rectangle maps, which could indirectly introduce other unrelated information.

Biermann et al. in (Biermann, Martin et al. 2002) enriched the Catmull-Clark subdivision schemes by developing a new method for preserving sharp features or defining creases. During the same time, they presented the algorithms for intuitive cut-and-paste editing on multi-resolution surfaces at interactive rates, which enhances the robustness of manipulation tools (Biermann, Martin et al. 2002). The limitation of the second work lies in building the appropriate correspondence between the identified target regions and chosen source surfaces.

Active research in multi-resolution subdivision-based modeling is outlined by Peter Schroder (Schroder 1999). The author depicts its distinct opportunities in wavelet extensions, remeshing, irregular subdivision, combined subdivision schemes, compression, and integration of modeling and simulation. The common existing MRSS modeling methods were surveyed in depth by Zorin in (Zorin 2006). The surveyed methods include: free-form editing, Boolean operations, non-smooth features and topologically adding complex details. This survey provides a global view of the development in MRSS modeling, which clarifies concrete innovations in different domains.

2.3.3 Control of Mesh Level Of Details (LODs)

Level Of Details (LODs) relates to the accuracy of a mesh to approximate a geometric model (De Floriani and Magillo 2002). The history of the LODs research follows the development of multi-resolution modeling.

LODs is used usually in rendering distant objects by simplifying the amount of details. The earlier applications and the recent approaches of LODs are summarized by Luebke et al in (Luebke, Reddy et al. 2002). This book comprehensively surveys various run-time LODs frameworks for managing level of details, such as selecting LODs (optimization-based predictive schedulers), discrete LODs (traditional decoupling simplification and rendering), continuous LODs (progressive meshes) and view-dependent LODs (vertex hierarchies). Besides, the algorithms and the operators for simplifying models, the error metrics for evaluating LODs creation, run-time management of selection criteria, popping prevention, frame rates and resources (ex. memory, internet streaming) are discussed both in theory and in practice.

In recent researches, LODs are widely used for interactively rendering terrain models. Representatively, Pajarola and Gobbetti reviewed the major LOD error metrics proposed in the related terrain triangulation algorithms: 1) object-space approximation error and 2) Image-space approximation error (Pajarola and Gobbetti 2007). In this survey, the authors stressed that the types of LOD error metrics have an important influence on structure sizes and efficiency of the triangulated mesh results.

2.3.4 General Overview of the Multi-resolution Modeling

The concept of multi-resolution is introduced for improving the precision and the efficiency of approximating the studied meshes. From above, multi-resolution modeling covers a large variety of research subjects. In the previous sections, mesh simplification, mesh refinement and LODs control were reviewed. In the multi-resolution framework, the works specifically related to mesh refinement are usually based on subdivision techniques. That is the reason why we present subdivision-based mesh refinement in a distinct section. Since the algorithms and the data structure used for representing multi-resolution meshes, LODs and remeshing techniques are also important issues in multi-resolution modeling, in this section, we review the related approaches.

The models and the data structures of spatial objects' multi-resolution mesh representations are presented in (De Floriani and Magillo 2002), which implies the continuous improvement and gradual maturity of multi-resolution technology. Later, the Adjacency and Incidence Framework (AIF) for dynamic multi-resolution meshing algorithms was proposed as an efficient data structure to manage multi-resolution meshes (Silva and Gomes 2003). In the work of Shaffer and Garland (Shaffer and Garland 2005), a powerful data structure with an efficient access to multi-resolution meshes and flexible operations on massive geometric data was built.

The publication of Level of Detail for 3D Graphics from the Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, supplied comprehensive references in the evaluation and optimization of multi-resolution meshes, which predicates the incorporation and future of multi-resolution modeling and LODs technology (Luebke, Reddy et al. 2002).

A multi-resolution surface representation of hybrid meshes, which combine the advantage of regular refinements allowing efficient data structures and the advantage of irregular operations allowing topology updating while keeping approximate detailed features in a hierarchy structure, is presented in the work of Guskov et al. (Guskov, Khodakovsky et al. 2002). The novelty of Guskov et al.'s work lies in constructing a multi-resolution surface representation for representing complex geometric hybrid meshes in a user-guided remeshing procedure, which allows to use irregular operations to keep topological and parametric changes on semi-regular meshes at multiple scales.

In the following years, as the interactive editing of multi-resolution meshes is improved, remeshing techniques (Alliez, Ucelli et al. 2005) with a comprehensive coverage of the issues (validity, quality, fidelity, discrete input, large data sets, uncertainty and correspondence), and with the consideration of algorithmic requirements in LODs, complexity and theoretical guarantees, have flourished in the multi-resolution modeling field.

Around 2006, Guskov proposed a semi-regular remeshing method of arbitrary shapes, which can be directly used for multi-level computational applications (Guskov 2006). This approach provides an important reference to mesh reconstruction and optimization analysis in the context of multi-level computations. Besides, the proposed parameterization procedure illustrates the comparison between the initial mesh and the modified meshes with extended features.

Recently, in the survey of Pajarola and Gobbetti (Pajarola and Gobbetti 2007), the authors analyzed the common semi-regular multi-resolution approaches in adapting rendered grid-digital terrain model complexity, according to the respective issues in real applications: data structures, triangulation algorithms, error metrics, dynamic scene management and rendering methods. In the conclusion of this survey, the authors restated that the semi-regular multi-resolution methods are the best choices for visualizing large scale height-field data sets.

2.4 Multi-resolution Subdivision Techniques in Engineering Applications

Since using subdivision surfaces to animate movie characters gained popularity, the Multi-Resolution Subdivision (MRS) techniques have been widely integrated in the applications, such as computer graphics and higher order surface reconstructions in biomedical fields. However, even though the popularity of MRS techniques continuously evolves, until recently the advantages of implementing MRS techniques have not been significantly discovered in engineering applications. Furthermore, among the rare MRS contributions for simulation-based designs, the approaches are limited to surface modeling and are not extended well to volumetric modeling.

2.4.1 Issues and Concerns of Employing MRS

Around 2001, Gonsor and Neamtu investigated the capability of implementing subdivision surfaces into the Boeing developed surface geometry modeling system (the Aero Grid and Panel System (AGPS)) (Gonsor and Neamtu 2001). In this technical report, the authors identified the essential requirements of the subdivision implementation from the engineering perspectives: interpolation ability and tangent plane continuity.

Through addressing the theoretical and practical issues that CAD and CAGD communities confront, the authors arrive with two-sided conclusions. One the one side, they confirm the potential benefits that incorporating the Catmull-Clark subdivision surfaces could offer, such as the feasibility of simulating unstructured meshes, the capacity of evaluating and manipulating surfaces, the sufficient tangent plane continuity of the studied subdivision, the ability of globally parameterizing subdivision surfaces, the robustness of subdivision representations in unifying

geometric design, mechanical analysis and numerical manipulation, and the support of local refinement. On the other side, they conclude that the existing subdivision techniques are not ready to be broadly utilized in the Boeing geometry system due to the following concerns: the inflexibility of subdivision surface parameterization, the insufficient studies of approximation accuracy, the unanswered questions about curvature continuity and behaviour of subdivision patches surrounding extraordinary vertices, and the understudied interpolation subdivision schemes.

2.4.2 Recent Approaches in Subdivision Surfaces

Regarding the recent approaches in subdivision surface for engineering applications, we must mention the works from Cirak et al. (Cirak, Ortiz et al. 2000), Persson et al (Persson, Aftosmis et al. 2006), and Burkhart et al. (Ito, Shih et al. 2009, Burkhart, Hamann et al. 2010), where the first two innovative works are practical resources for our implementation of the subdivision parameterization techniques.

First, the contribution of Cirak et al. demonstrates that the finite element solution can be directly supported by the nodes representing the geometry. Usually, the Finite Element Method (FEM) uses purely local basis functions, which means the calculation of each face on a triangular surface mesh is only based on the three nodes within the triangle. When there is a displacement on a surface mesh, the changes on the deformed mesh bring up certain enforcing energy. In FEM, the enforcing energy must be spread over relatively isolated nodes, where a bending strain energy function is absolutely required. However, until now a right bending function has not been available for complex models; resorting only to FEM may cause spurious oscillations of the approximate solution (see Figure 2.15).

Cirak et al. in their work (Cirak, Ortiz et al. 2000) proposed using subdivision surfaces for a finite-element treatment of thin-shell equations describing the mechanical behaviour of the modeled structures. The enforcing energy was spread over initial continuous control meshes based on subdivision surfaces and a pre-node displacement vector was added to each control mesh node. This subdivision-based treatment has shown to achieve versatility and high accuracy.

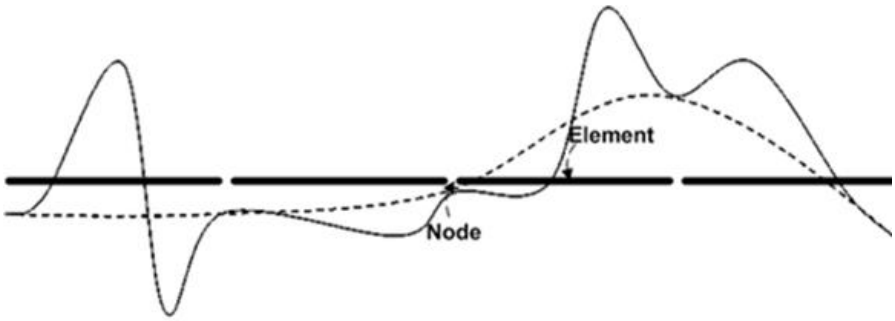

Figure 2.15: The illustration of spurious oscillations of the approximation solution (Cirak, Ortiz et al. 2000).

Persson et al. in their work (Persson, Aftosmis et al. 2006) examine using the Loop subdivision surface as surrogate geometry for CAD or analytic-based applications. The authors develop an algorithm to directly evaluate surface coordinates, derivatives, and curvatures at any arbitrary location on subdivision surfaces by integrating Stam's evaluation procedure (Stam 1998). The authors utilize the obtained curvature information as an indicator to adapt triangular mesh refinements on the finite-element analytic models. The results from this work demonstrate that the adaptive surface triangulations achieve higher-order accuracy rapidly and robustly.

Recently, Burkhart et al. have been particularly active in solid subdivisions. Until now, they have developed two types of solid subdivision schemes: one is based on $\sqrt{3}$ subdivision, which can be used for generating tetrahedral subdivision meshes (Burkhart, Hamann et al. 2010); and another one is based on Catmull-Clark subdivision, which can be used for generating hexahedral subdivision meshes (Burkhart, Hamann et al. 2010). The authors implemented respectively these two types of subdivision solids in FEA applications (Burkhart, Hamann et al. 2011). The works of Burkhart et al. remarkably demonstrate that solid subdivision representations can be employed for geometric design and computational analysis to bridge the gap between CAD and CAE (Computer Aided Engineering).

2.5 Restatement of Research Proposal

On the one hand, the literature review above indicates that subdivision-based representations can be utilized not only for representing the underlying geometry of complex physical models with

arbitrary topology, but also naturally support refinement operations on different dimensional meshes, such as surface and volume meshes. These two facts make subdivision-based representations really advantageous in unifying mesh and geometry for the numerical simulation in the context of engineering analysis. On the other hand, the related researches reveal that the current subdivision techniques have their limitations and unsolved problems in surface parameterizations, solid modeling, adaptive subdivision in the context of multi-resolution and implementations in engineering applications.

2.5.1 Restatement of Problems

From above, it is evident that the Loop subdivision scheme has already been successfully integrated in surface modeling for generating smooth limits with the desirable surface continuity C^2. But its implementation has not been truly completed in solid modeling. Besides, Loop subdivision in essence is an approximation subdivision scheme. This fact limits its integration with current numerical analysis methods.

As we have discussed in Section 2.1.3, the subdivision limits can be directly calculated by using the subdivision surface parameterization techniques. In practice, Stam's evaluation matrix (Stam 1998) is commonly used for calculating limit positions of arbitrary vertices on Loop subdivision surfaces. However, the triangle where the calculated surface vertex lies is limited to contain only one extraordinary vertex. This limitation indicates that Stam's evaluation method is not able to calculate the subdivision limits, when their control meshes include triangular faces with crease definitions.

The introduction of multi-resolution modeling clarifies that a multi-resolution subdivision mesh is a collection of multiple subdivision mesh parts, whose LODs vary. Traditionally, it requires a multi-level treelike structure to store various mesh parts individually obtained from different subdivision levels. This hierarchical data structure is obtained by using a non-uniform multi-level refinement method. This multi-level refinement method is based on the fact that different mesh densities of subdivision mesh parts can be obtained by correspondingly applying subdivision-based refinements different times. Thus, a multi-resolution subdivision mesh can be reconstructed by stitching together the subdivision mesh parts at multiple subdivision levels. However, in real implementation, this multi-resolution mesh generation method brings up a lot of practical issues: 1) it requires information to access the different subdivision mesh parts at the

different subdivision levels to be stored. Consequently, the storage could be very heavy, particularly when the subdivided meshes are volumetric meshes; 2) stitching the different mesh parts together could be very complicated, since it is very difficult to match the topology of the boundaries of different mesh parts with various LODs. This problem becomes even more intractable when mesh parts are volumetric subdivision meshes.

2.5.2 Restatement of Our Refinement Works

We propose the use of subdivision-based representations based on multi-resolution Loop subdivision limits, which fundamentally involve three techniques: the solid Loop subdivision scheme, Loop subdivision limits and the multi-resolution mesh generation.

Firstly, a new solid subdivision based on the Loop subdivision scheme is developed in the framework of refining tetrahedral meshes. This solid subdivision can be used for generating tetrahedral meshes with smooth boundary surfaces. Our solid Loop subdivision rules are used for updating the existing and the newly inserted vertices in interior volumes and boundary surfaces, together with preserving sharp features on subdivision solids. The mesh connectivity of tetrahedral meshes is going to be particularly studied, since it is an important factor influencing algorithm complexity. Besides, the mesh quality of newly generated tetrahedral subdivision meshes is going to be evaluated.

Secondly, the parameterization techniques on subdivision surfaces are going to be further studied. They are used for projecting the updated vertices on boundary surfaces to their boundary limits. The limitation of Stam's work is going to be removed through our study.

Thirdly, we introduce the single-level refinement method, which is developed for merging adaptive refinements and exact evaluations on boundary subdivision surfaces. This method is specifically introduced to overcome the drawbacks of uniform subdivisions and expensive hierarchical information storage in the non-uniform multi-level refinement method. In addition, we demonstrate that our single-level refinement method can be applied for refining curves, surfaces and volume meshes.

Fourthly, we present the algorithms used for generating multi-resolution subdivision meshes, particularly related to tetrahedral subdivision mesh generation. We demonstrate the refined mesh results obtained by applying the analytic-based criteria in the context of numerical simulations.

Finally, we present the adaptively refined tetrahedral mesh results with crease definitions. Besides, we are going to analyze system performance by evaluating the time costs of mesh connectivity calculation, new tetrahedron creation and limit position calculation.

CHAPTER 3
SOLID SUBDIVISION BASED ON THE LOOP SCHEME

The subdivision-based refinement method, composed of two essential elements: subdivision and adaptivity, is proposed. This method can be used to refine curves, triangular surface meshes and volumetric tetrahedral meshes. To address current challenges of integrating subdivision techniques into modeling volumetric meshes, a new solid subdivision is developed.

This chapter is organized into three sections. In the first section, we present one new solid subdivision scheme based on the Loop scheme, which includes: 1) topological splits (1-to-8 tetrahedral splits) based on full tetrahedral edge bisections and 2) geometric smoothing on tetrahedral boundary meshes integrated with the standard Loop scheme's boundary stencils. We enrich the geometric smoothing rules with four standard cases and sharp feature preservation algorithms. At the end of the first section, we summarize the advantages of this new solid Loop subdivision scheme by comparing with Chang's solid subdivision work. In the second section, firstly we present the challenges of implementing this new solid subdivision scheme in a traditional refinement way for real applications. Then we propose using adaptive subdivision to meet the challenges in practice. At the end of this chapter, we present the solution of integrating adaptivity support in solid subdivision.

3.1 Loop-based Solid Subdivision Scheme

A volume mesh can be topologically viewed as two parts: the boundary and the interior. For a tetrahedral mesh, the boundary is a shell surface composed of triangular faces, and the interior is filled with tetrahedrons. The basic geometric elements of any tetrahedral mesh are *Vertex* (V_i), *Edge* (E_i), *Face* (F_i) and *Tetrahedron* (T_i).

3.1.1 1-to-8 Tetrahedron Split

Loop subdivision is an edge bisection scheme. One tetrahedron has 4 vertices (v_1, v_2, v_3, v_4) and 6 edges $E_1(v_1, v_3), E_2(v_2, v_3), E_3(v_1, v_4), E_4(v_2, v_4), E_5(v_3, v_4)$ and $E_6(v_1, v_2)$ (see Figure 3.1(a)). A full tetrahedron split is achieved by inserting one new vertex in the middle of each edge, which introduces 6 new edge vertices e_5, e_6, e_7, e_8, e_9 and e_{10}. After splitting, there are three connection

choices for the internal middle edge: (e_7, e_9), (e_6, e_8) or (e_5, e_{10}). To obtain appropriate mesh quality during splitting, the edge with the shortest length is chosen. Thus, one full tetrahedron split generates 8 smaller tetrahedrons, which is denoted as 1-to-8 tetrahedron split as shown in Figure 3.1(b). In the tetrahedral example of Figure 3.1, e_7 and e_9 is internally connected.

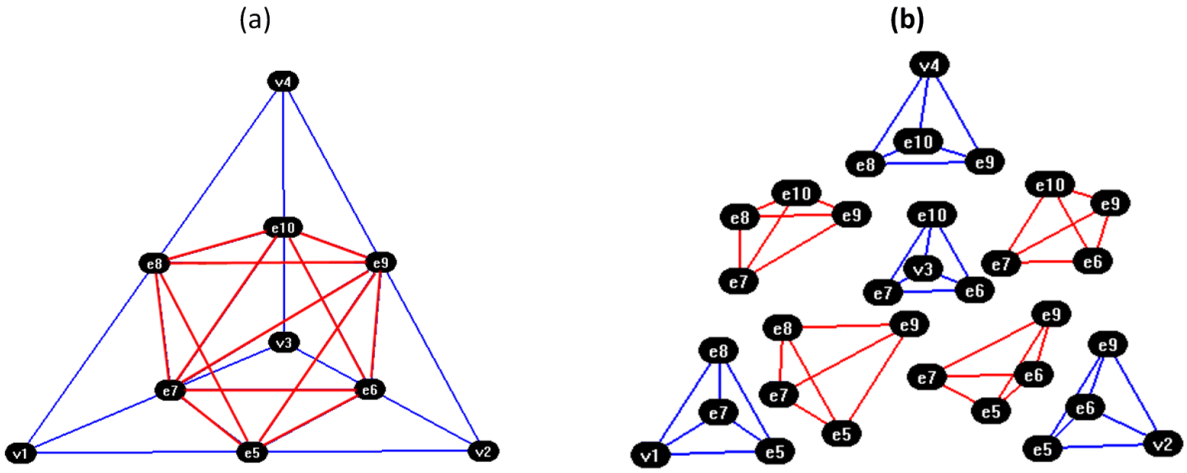

Figure 3.1: (a) Tetrahedron split with four original tetrahedral corner vertices (v_1, v_2, v_3, v_4) and 6 new edge vertices $e_5, e_6, e_7, e_8, e_9, e_{10}$. (b) Edge bisection of a tetrahedron into 8 smaller tetrahedrons.

The process of iteratively applying the Loop subdivision-based scheme on a Schoenhardt tetrahedral mesh is shown in Figure 3.2. This Schoenhardt model comes from Tetgen: a quality tetrahedral mesh generator and a 3D Delaunay triangulator, which is developed by Hang Si (Hang). From the refinement results, we can observe that the total number of the new tetrahedrons generated in the volume mesh is increased by a factor of eight after one subdivision iteration. Figures 3.2(a.1) and 3.2(b.1) present the initial control mesh of the Schoenhardt example. Respectively, the result obtained after once subdivision is displayed in Figures 3.2(a.2) and 3.2(b.2)... etc.

As the subdivision level is increased, the boundary surface of the corresponding subdivided mesh converges toward a smooth surface, and the shape of this boundary surface shrinks toward

the interior of the boundary surface of its initial control mesh. The mesh density is correspondingly increased, resulting from topological splits and geometrical smoothing.

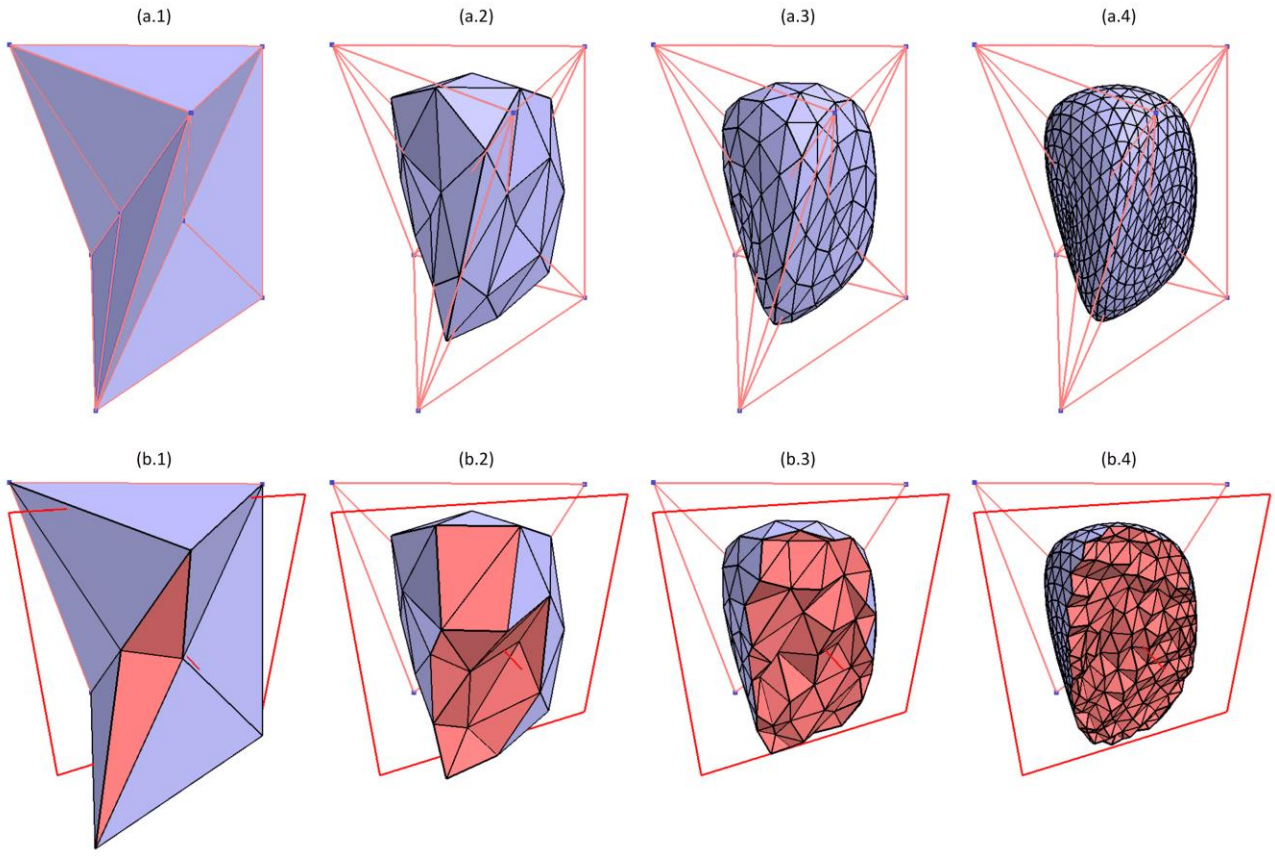

Figure 3.2: Iteratively applying the Loop-based solid subdivision on the Schoenhardt example: (a.1)-(a.4): their exterior profiles at multiple subdivision levels; (b.1)-(b.4): their profiles displayed with cut planes.

3.1.2 Solid Geometric Smoothing Rules

Geometric smoothing consists in updating the geometric positions of 1) the existing tetrahedral vertices; 2) the inserted edge vertices. These geometric smoothing rules can further be classified as being applied to: a) vertices on boundary surfaces and b) vertices in interior volume. We summarize them into four standard cases: (1a) existing boundary corner vertex; (1b) existing interior corner vertex; (2a) inserted boundary edge vertex; (2b) inserted interior edge vertex (see Figure 3.3). In the following sub-section, we detail smoothed position calculations of these four cases.

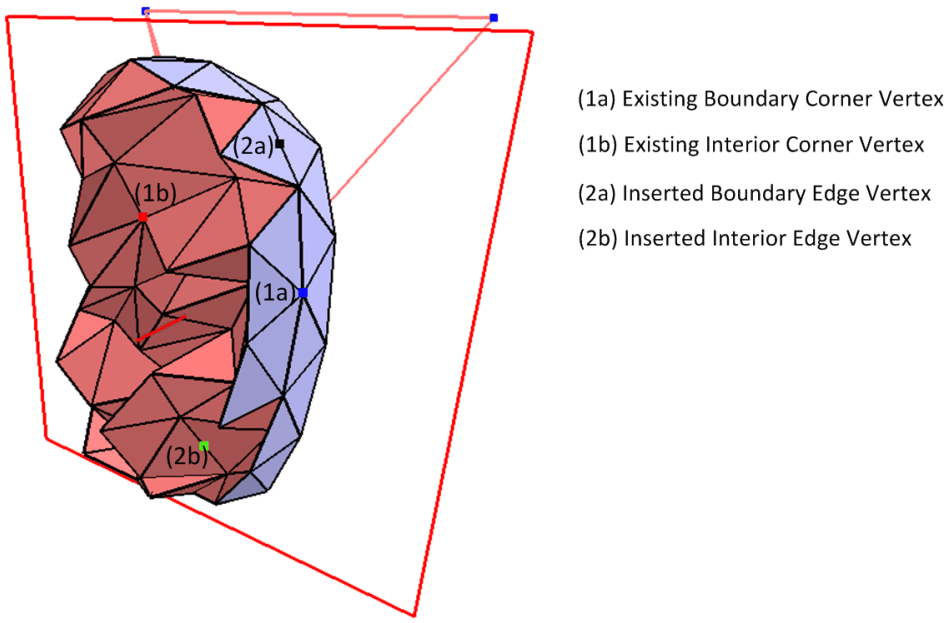

Figure 3.3: Four standard cases on the Loop subdivision-based tetrahedral mesh of the Schoenhardt example displayed with a cut plane.

3.1.2.1 The Four Standard Cases

To calculate the smoothed position of an existing vertex lying at a boundary corner position, we need its former positions and the positions of its neighbouring boundary vertices at the lower subdivision level. The case(1a) calculation is based on the Loop subdivision surface mask of an even interior vertex (see Equation 2.10, Equation 2.12 and Figure 2.6(b) in Section 2.1.3).

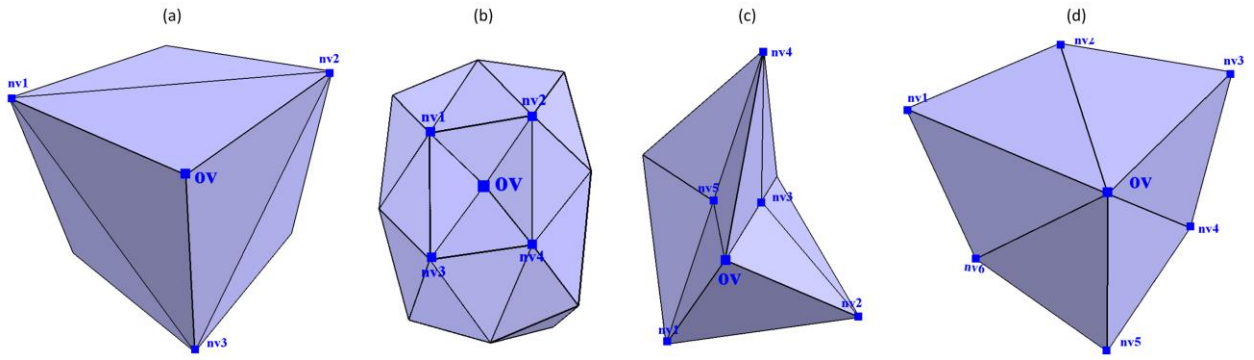

Figure 3.4: Various edges sharing the same existing vertex on boundary surfaces: (a) and (d): one cube mesh example at the initial subdivision level; (b) the Schoenhardt mesh subdivided once; (c) the Schoenhardt mesh at the initial subdivision level.

An existing boundary corner vertex is denoted as ov and its direct neighbouring vertices as nv_i (see Figure 3.4). The corresponding weights c to the involved vertices vary according to the number of edges num on boundary surfaces directly sharing the existing vertex ov. The new position of the existing boundary vertex new_ov at higher subdivision level can be obtained by using Equation 3.1:

$$new\_ov = \begin{cases} \frac{7}{16}\cdot ov + \frac{3}{16}\sum_{i=1}^{3} nv_i & (num = 3), \\ \frac{5}{8}\cdot ov + \frac{3}{8\cdot num}\sum_{i=1}^{num} nv_i & (num > 3). \end{cases} \tag{3.1}$$

To calculate the new position of an existing corner vertex in interior volumes, we average its former position and the positions of its neighbouring vertices at the lower subdivision level. Here, we denote one existing interior corner vertex as iv and its direct neighbouring vertices as nv_i (see Figure 3.5(a)). As opposed to existing corner vertices on boundary surfaces, the neighboring vertices of iv include all the directly connected vertices, which can be interior or on boundary surfaces. More specifically, the topological neighbouring connectivity around one interior vertex is illustrated in Figure 3.5(b). The number of its direct neighbour vertices is denoted as num.

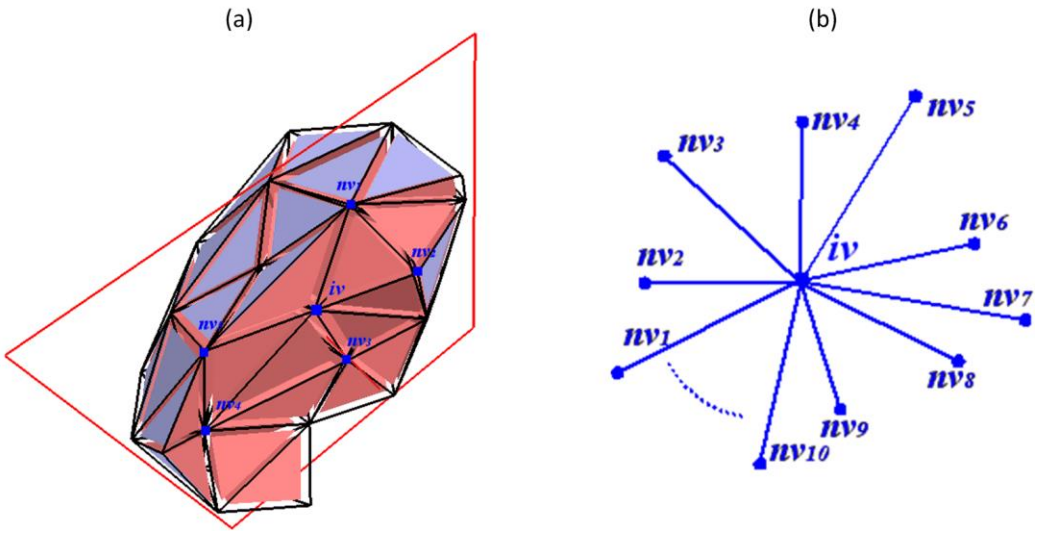

Figure 3.5: Topological neighbouring connectivity around an existing interior vertex.

The geometric smoothing on boundary surfaces leads to surface shrinkage towards the interior, which is taken into consideration when computing the geometric averaging. The corresponding averaging operation is controlled by using weights which sum up to one unity. The influence of the existing interior corner vertex at the lower subdivision level to the position at the higher subdivision level is accounted by using a weight c. In our research, we remark that the concrete weight values change the shape of the relevant interior tetrahedrons. And this fact could consequently determine the mesh quality of new generated tetrahedral meshes. Due to the limitation of our research scope, we simply assign a fix value, such as '0.8' to c. This weight value is adjusted by visualizing split results and considering that the former existing position majorly determines the new position new_iv. The smoothed position of one existing interior vertex iv in the case (1b) is calculated using Equation 3.2:

$$new\_iv = c \cdot iv + \frac{(1-c)}{num} \sum_{i=1}^{num} nv_i. \tag{3.2}$$

To calculate the position of the newly inserted boundary edge vertex, we apply the Loop subdivision surface stencil of the odd interior vertex (see Equation 2.9 and Figure 2.6(a) in Section 2.1.3). In order to clarify the relationship between the newly inserted edge vertex and its

neighbouring vertices on boundary surfaces, we illustrate the calculation of the case (2a) on different mesh examples (see Figure 3.6).

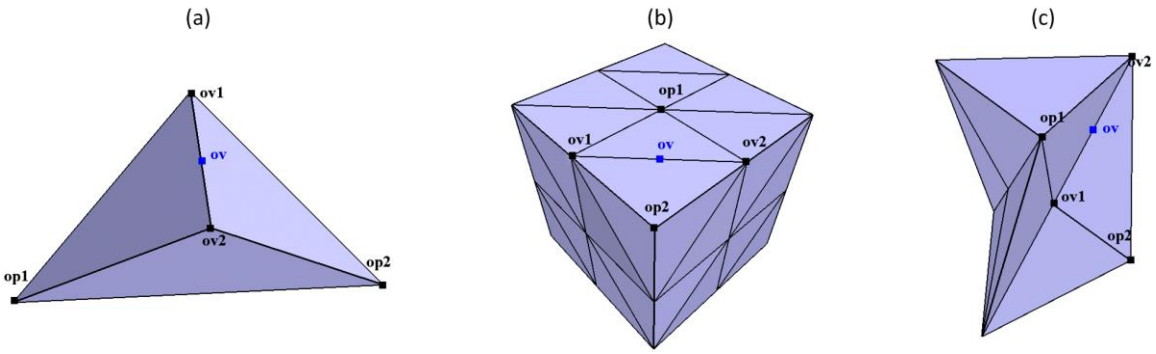

Figure 3.6: Calculation of a newly inserted edge vertex on boundary surfaces of different tetrahedral mesh examples: (a) a tetrahedron example; (b) a cube example; (c) a Schoenhardt tetrahedral mesh.

Here, we denote one newly inserted boundary edge vertex as ov, its two direct neighbouring vertices sharing the same split boundary edge as ov_1 and ov_2, the other two neighbouring vertices in the neighbouring boundary triangles sharing the same split boundary edge as op_1 and op_2. The position of the newly inserted boundary edge vertex ov can be calculated by using Equation 3.3.

$$ov = \frac{3}{8}ov_1 + \frac{3}{8}ov_2 + \frac{1}{8}op_1 + \frac{1}{8}op_2. \tag{3.3}$$

We integrate the subdivision mask of an odd boundary vertex into the calculation of a newly inserted edge vertex in interior volumes (see Equation 2.13 and Figure 2.7(a) in Section 2.1.3). We demonstrate the calculation of the case (2b) on different tetrahedral mesh examples (see Figure 3.7).

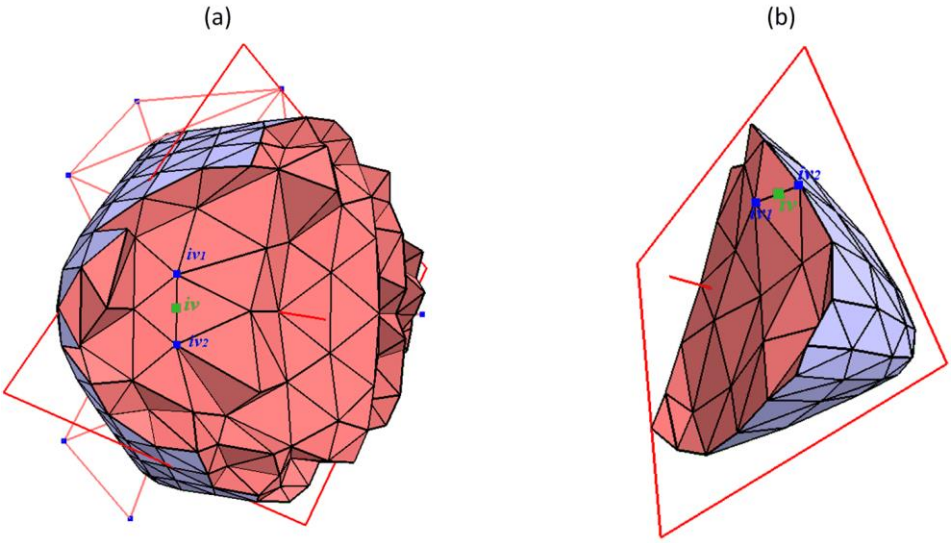

Figure 3.7: Calculation of a newly inserted interior edge vertex on different tetrahedral mesh examples: (a) a cube example; (b) a tetrahedron example.

Here, we denote the two endpoints of the split interior edge as iv_1 and iv_2 and the newly inserted interior edge vertex as iv. The calculation of the case (2b) is expressed in Equation 3.4, which is to simply obtain the middle position of the split interior edge:

$$iv = \frac{1}{2} iv_1 + \frac{1}{2} iv_2. \tag{3.4}$$

3.1.2.2 Crease Definitions on Boundary Surfaces

As the subdivision level increases, boundary surfaces of tetrahedral meshes are gradually smoothed. In solid modeling, for the purpose of conforming to the physical criteria of CAD models, real applications require local sharp features on boundary surfaces, such as high curvatures or sharp edge features, to be preserved during geometric smoothing iterations. In order to meet these special requirements, we introduce a new formulation that allows us to integrate boundary crease creation into the Loop-based solid subdivision scheme.

In the present work, we develop two types of boundary crease creation: 1) boundary vertex creases and 2) boundary edge creases.

3.1.2.2.1 Boundary Vertex Creases

Boundary vertex crease creation consists in keeping the selected boundary vertices unchanged during subdivision-based smoothing. More specifically, the vertices on boundary surface tagged as “boundary vertex creases” remain at the same positions as the ones at their current tagged subdivision levels, while at the same time, their neighbouring boundary vertices are relocated approximately according to the corresponding subdivision stencils. Here, we illustrate an example of defining a boundary vertex crease in Figure 3.8 and another example of defining multiple boundary vertex creases in Figure 3.9.

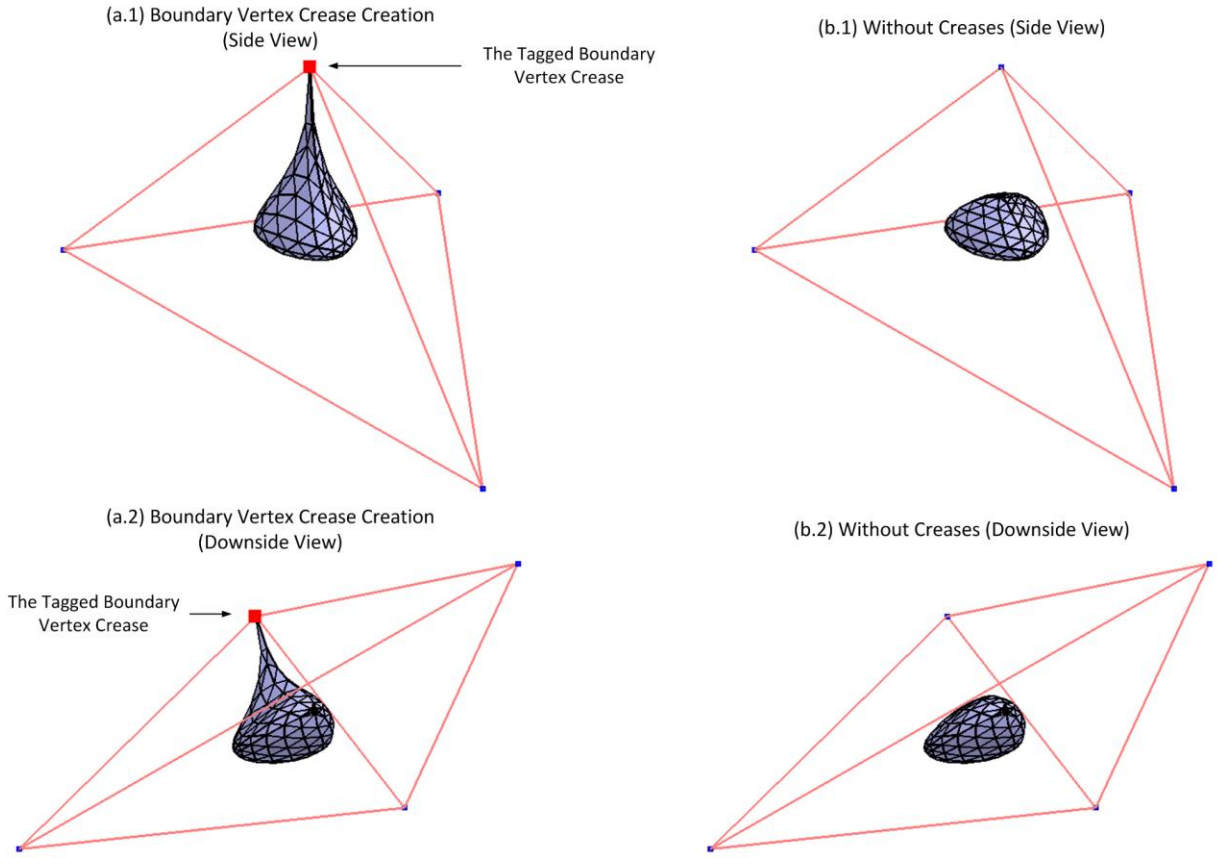

Figure 3.8: Defining a boundary vertex crease on a tetrahedron example: (a.1-a.2) the result of applying subdivisions three times with defining the selected vertex as the boundary vertex crease; (b.1-b.2) the result of applying subdivisions three times without defining boundary vertex creases.

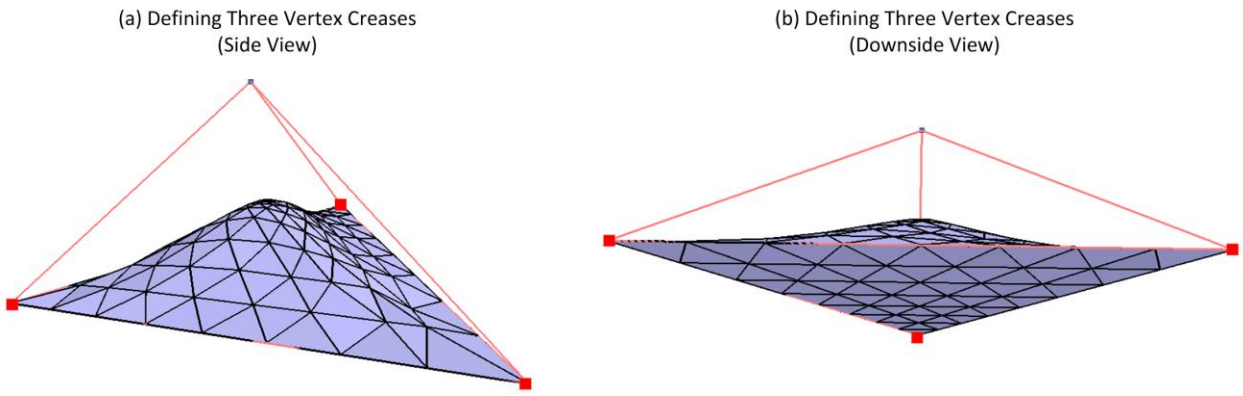

Figure 3.9: Three vertex crease definitions (displayed as large red vertices) on a tetrahedron boundary surface. Illustrating the results of applying subdivisions three times with vertex crease definitions from different views in (a) and (b).

3.1.2.2.2 Boundary Edge Creases

Boundary edge crease creation consists to tag the boundary surface edges of volume meshes as creases, and smooth the tagged edges as the isolated curves instead of applying subdivision surface operations on whole boundary surfaces. To demonstrate the difference between boundary edge creases and boundary vertex creases, we illustrate the result of creating multiple edge creases on the same tetrahedron example (see Figure 3.10).

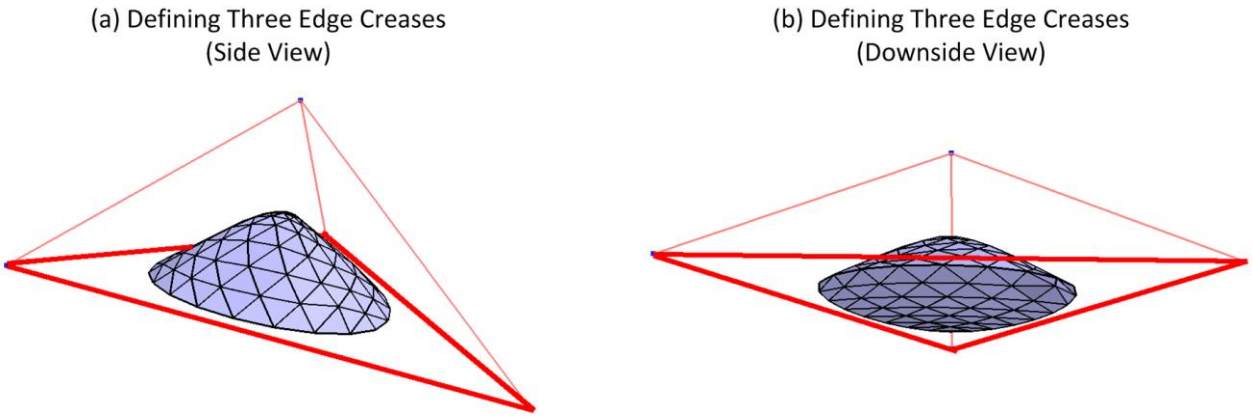

Figure 3.10: Three edge crease definitions (displayed as large red lines) on a tetrahedron boundary surface. Illustrating the result of applying subdivision three times with edge crease definitions from different views in (a) and (b).

In our work, boundary edge creases can be defined anywhere as desired by selecting and labelling any boundary surface patch or edge curve on the required tetrahedral mesh parts. Here, we demonstrate another example of boundary edge crease creation on a CAD model: an unstructured tetrahedral mesh of a gear (Hang). Its initial control mesh contains 294 control vertices and 698 tetrahedrons (see Figure 3.11(a)). In this example, we define the boundary edge creases on the upper part and the lower center part of the gear example to keep the corresponding sharp features. The boundary edge selection on the initial control mesh is illustrated in Figure 3.11(a) and the results of preserving the involved sharp features are presented in Figure 3.11(c).

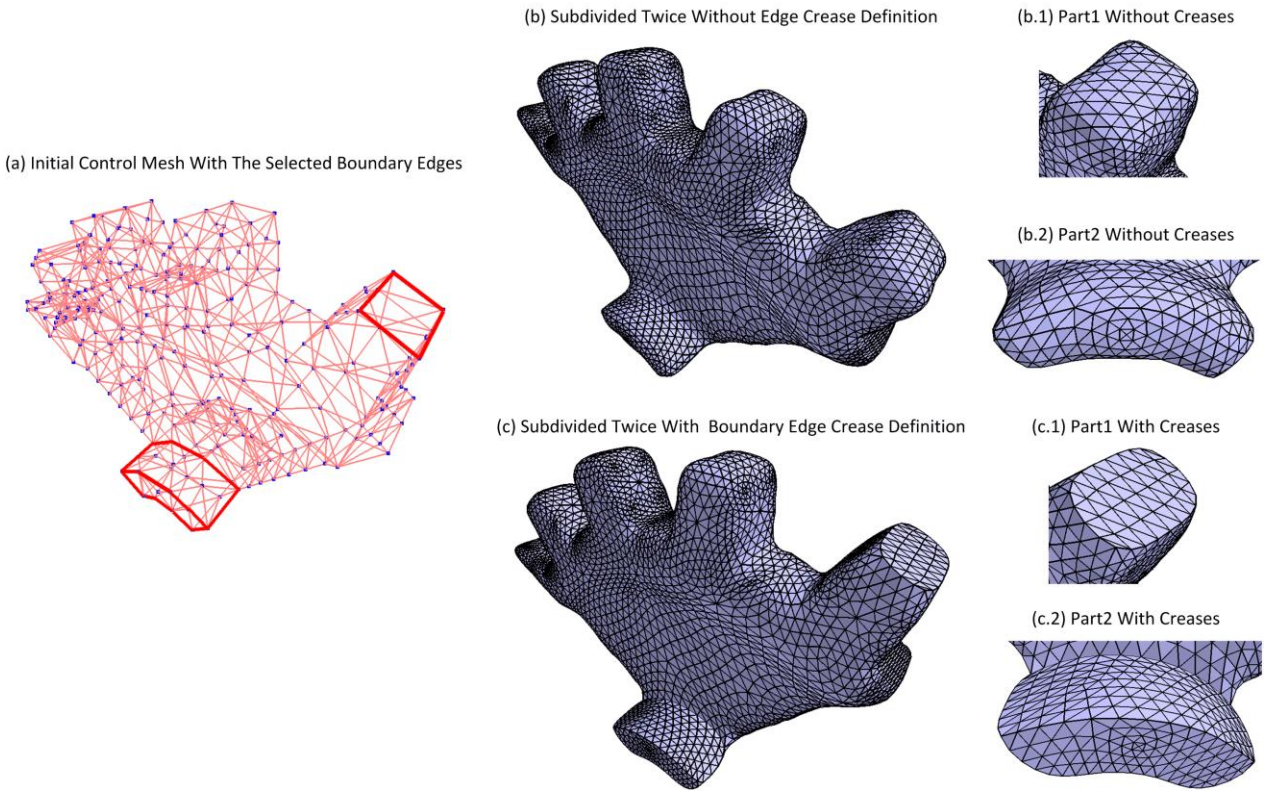

Figure 3.11: Boundary edge crease creation on an unstructured tetrahedral mesh of a gear: (a) defining the boundary edges as creases on the initial control mesh, where the edge creases are marked with thick red lines; (b) applying solid subdivision iterations without defining creases; (c) the subdivision results with boundary edge crease definition.

3.1.2.2.3 Special Cases in Boundary Edge Crease Creation

When two or more than two boundary edges sharing the same boundary vertex are tagged as boundary edge creases, we classify them into two special cases according to their particular topology: Case A and Case B. Case A: two boundary edge creases are connecting with the same existing boundary vertex (see Figure 3.12 (a.1)). Case B: more than two boundary edge creases are sharing the same existing boundary vertex (see Figure 3.12(b.1)).

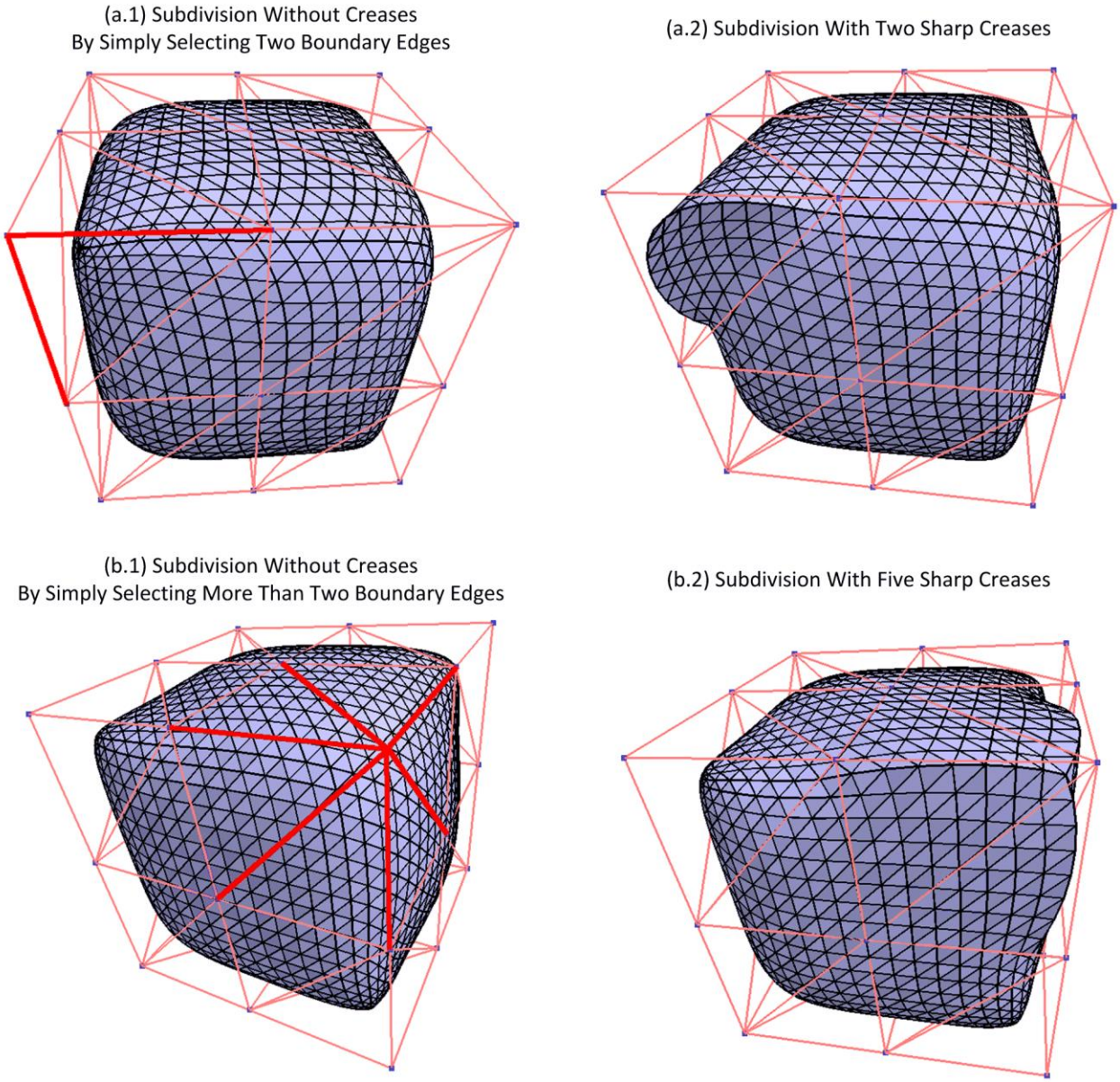

Figure 3.12: Defining two (case A) or more than two boundary edge creases (case B) on a tetrahedral mesh of a cube.

The stencil used for calculating Case A is referenced from the regular Loop subdivision surface mask of an even boundary vertex (see Equation 2.14 and Figure 2.7(b)). For Case B, we developed a new stencil based on the regular Loop subdivision mask of an even interior vertex (see Equation (2.10), Equation (2.12) and Figure 2.6(b)) by treating all neighbouring boundary edges as boundary creases. The pseudo code for calculating the new position of the boundary vertex shared by two or more than two boundary edge creases is detailed in Table 3-1.

Table 3-1: The pseudo code for calculating the new position of the boundary vertex shared by two or more than two boundary edge creases

[Special Cases]: The boundary crease vertices in Case A and Case B

[Input]: The former position of the existing boundary vertex ov, the list of ov's neighbouring boundary vertices $list\_bv$, the list of neighbor boundary creased vertices $list\_bcv$, the current subdivision mesh $Mesh$

[Output]: The new position of the existing boundary vertex new_ov

```
function GetNewPosition(Mesh, ov)
{
        if (list_bcv.size == 2)
        {
                new_ov = list_bcv[0] * 1/8 + list_bcv[1] * 1/8 + ov * 6/8;
        }
        else if (list_bcv.size > 2)
        {       base = 1;
                if (list_bv.size == 3)
                {
                        weight = base/16;
                }
                else
                {
                        weight = base/(8 * list_bv.size());
                        new_ov = ov * (1 - weight * list_bv.size());
                        for each vertex v in list_bv
                        {
                                new_ov += v * weight;
                        }
                }
        }
}
```

3.1.3 Summary of this Loop-based Solid Subdivision Work

From above, recursively applying solid Loop subdivisions on tetrahedral meshes generates multiple subdivision-based tetrahedral meshes at different LODs. The solid subdivision scheme developed in our work combines 1-to-8 topological split with the Loop subdivision scheme-integrated geometric smoothing. The implementation results show that the boundary surfaces of the subdivided tetrahedral meshes are gradually smoothed as the subdivision levels are correspondingly increased.

By comparison with the solid subdivision scheme based on Box Splines proposed in Chang's work (Chang, McDonnell et al. 2002), the advantages of this approach can be summarized as: 1) 1-to-8 tetrahedron split only generates tetrahedrons, whereas the topological splits in Chang's work introduce octahedral forms. Thus our topological splitting method simplifies the relevant data structure, which permits to work with any tetrahedral mesh with arbitrary topology. Moreover, this fact is favorable in numerical analysis, since it is necessary to have only one type of volumetric element for most numerical methods. 2) Our geometric smoothing algorithms can be simply divided into the four standard cases and the special cases related to boundary vertex or edge creases. Our key contribution here is to extend the Loop subdivision scheme from smoothing surface meshes to smoothing boundary surfaces of 3D tetrahedral meshes.

Moreover the sharp feature preservation function developed in our work, can be naturally integrated into subdivision-based smoothing procedures. This integration is one additional advantage of utilizing subdivision-based representations to unify geometry smoothing and mesh generation. The reason is that, by contrast, in traditional volumetric B-rep modeling, sharp feature preservation is carried out in extra procedures, such as extracting, detecting, preserving and eliminating functions, which could involve complex algorithms (Qian and Zhang 2010).

3.2 Practical Implementation of the Loop-based Solid Subdivision Scheme

The implementation of the proposed solid subdivision scheme in real applications encounters two common challenges: precision and efficiency. As addressed before, the boundary surface continuity of subdivision limits represents attractive features for engineering applications. In our

refinement work, we use tetrahedral subdivision limits to represent the true geometry of the studied solid models.

3.2.1 Uniform Approximating Iterations

In this research stage, multiple subdivided tetrahedral meshes at different LODs are generated from iterative subdivision-based refinements, where the approximations generated during the first several iterations are always far from their subdivision limit. And only after a few iterations, the refined tetrahedral meshes will be close to this target 'true' geometry. Usually, boundary surface profiles of volumetric meshes in 2D can be simulated as curves. Here, in order to better visualize this gradual approaching procedure, we illustrate applying the traditional subdivision-based refinements on a square curve example. The initial curve mesh M^0 is a collection of four edges (the square curve) and consists of four initial control vertices $\{p_0^0, p_1^0, p_2^0, p_3^0\}$. The interior approximating circle curve is the subdivision limit L and the limit positions of the four initial control vertices $\{p_0^\infty, p_1^\infty, p_2^\infty, p_3^\infty\}$ are correspondingly projected on L (see Figure 3.13(a)). The approximation results generated from the traditional subdivision iterations are shown in the examples (b.1-b.3) of Figure 3.13. From these examples, we can observe that the refined mesh M^j at the j^{th} subdivision level shrinks towards the subdivision limit L (the interior approximating circle curve) through iteratively applying the Loop subdivision rules on the refined curve meshes.

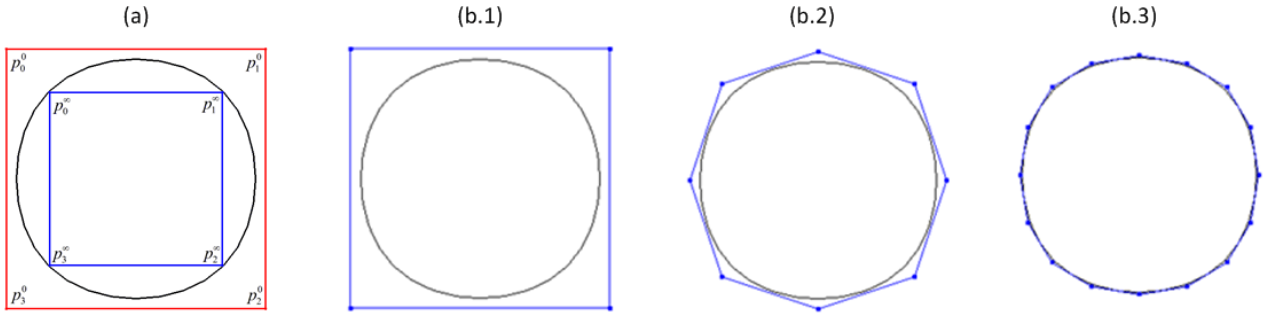

Figure 3.13: The approaching procedure of traditional subdivision iterations.

The approaching procedure above can be summarized as an unlimited approximation procedure. More specifically, the refined results are getting closer and closer but never exactly on the true geometry. Besides, this procedure is based on uniform iterations, which could expose the system to tedious unnecessary refinements in some regions. Evidently, edge bisection-based splits extended from triangular faces to tetrahedral elements increase, not only the complexity of mesh topology, but also the expansion of geometrical data. A heavy burden is imposed on system resources particularly when the iterations are applied completely on volumetric meshes until the desired subdivision level is achieved.

3.2.2 Eight-child Tree Structure

A global data structure is constructed to keep the hierarchy of multiple subdivision-based tetrahedral meshes at various resolutions and to support the transmissions among different subdivision iterations. The basic element used through the global data structure is a tetrahedral mesh M^j at the j^{th} subdivision level, which is a collection of explicit sub-elements: vertices (V^j), faces (F^j) and tetrahedrons (T^j) (see Table 3-2).

Table 3-2: The sub-elements in a tetrahedral mesh M^j

```
class Mesh
{
    VECTOR(VertexPointer) Vertices;
    VECTOR(FacePointer)   Faces;
    VECTOR(TetraPointer)  Tetras;
    …
};
```

The sub-element vertex V^j contains the position of the current vertex and hierarchy information to link to parent or child vertices. The ParentVertex contains two pointers or only one, depending on whether the parent vertex is a newly inserted vertex or an existing one. The ChildrenVertex is used when the descendants of an existing vertex need to be tracked. The information of fixed vertex crease or edge crease definition is included in the class of vertex for further propagating creases on boundary surfaces. In addition, to achieve proper topological

splitting, we add the neighbouring vertex information, which is identified according to corresponding types: standard, boundary or boundary crease. The description of the sub-element vertex is detailed in Table 3-3.

Table 3-3: The description of the sub-element vertex V^j

```
class Vertex
{
        private:
        Vector3d pos; // position

        //hierarchy relationship
        VertexPointer ParentVertices[2];
        VertexPointer ChildrenVertex;

        //connectivity at the current subdivision level
        VECTOR(VertexPointer) NeighborVertices;
        VECTOR(VertexPointer) NeighborBoundaryVertices;
        VECTOR(VertexPointer) NeighborBoundaryCreaseVertices;

        bool edgecrease; //edge crease
        bool fixed; // vertex crease
};
```

The basic definition of a face sub-element F^j is based on its three vertices. The hierarchy information is carried through by the parent face from which the current face comes and the four child faces generated by each split. The connectivity information is constructed by its values of NeighborBoudaryFaces and NeighborTetras at the same subdivision level. If the number of neighbouring triangular faces is equal to three, we remark the current face to be on one boundary surface, which we use NeighborBoundaryFaces to indicate. To differentiate interior tetrahedrons from the others on boundary surfaces, we use NeighborTetras to inform how many tetrahedrons are sharing the given face. If the NeighborTetras value equals one, it indicates the current face is on one boundary surface, whereas if the NeighborTetras value equals two, it indicates the current face is in the volume interior. The description of the face sub-element is detailed in Table 3-4.

Table 3-4: The description of the sub-element face F^j

```
class Face
{
        private:
        VertexPtr Vertices[3]; // position

        //hierarchy relationship
        FacePtr ParentFace;
        FacePtr ChildrenFaces[4];

        //connectivity at the current subdivision level
        FacePtr         NeighborBoundaryFaces[3];
        TetraPtr        NeighborTetras[2];

        bool boundary;
};
```

The basic sub-element for topological splitting in the proposed mesh refinement operations is tetrahedron T^j. Its geometrical definition consists of four vertices and four triangular faces. About hierarchy information, we use ParentTetra to trace where its parent tetrahedron is, and we use ChildrenTetra to indicate where its eight child tetrahedrons are. In terms of mesh connectivity, the number of the neighbouring tetrahedrons of a tetrahedron may vary from 0 to 4. Since we apply the geometric smoothing operation only on the boundary surface vertices, it is necessary to obtain the information of whether the current tetrahedron is on a boundary surface. More specifically, when the number of the neighbouring tetrahedrons is less than 4, we mark the current tetrahedron as boundary. The description of the sub-element tetrahedron is detailed in Table 3-5.

In brief, during 1-to-8 iterative subdivisions, one tetrahedron is subdivided into 8 smaller tetrahedron descendants, and each tetrahedron descendant is subsequently split into 8 other child tetrahedrons. Subdivisions are recursively continued until the refined tetrahedral mesh approaches the true geometry as required. We use an eight-child tree (Wikipedia.org) to describe this hierarchical structure linking all subdivision levels, where a root node represents the initial control tetrahedral mesh and the child nodes represent the consecutive subdivision levels (see Figure 3.14).

Table 3-5: The description of the sub-element tetrahedron T^j

```
class Tetra
{
        private:
        VertexPtr Vertices[4];
        FacePtr Faces[4];

        //hierarchy relationship
        TetraPtr ParentTetra;
        TetraPtr ChildrenTetras[8];

        //connectivity in same level
        TetraPtr NeighborTetras[4];

        bool boundary;
};
```

Evidently this eight-child tree structure requires storing all connectivity information which supports navigations between different subdivision levels, which is complicated and heavy on system resources.

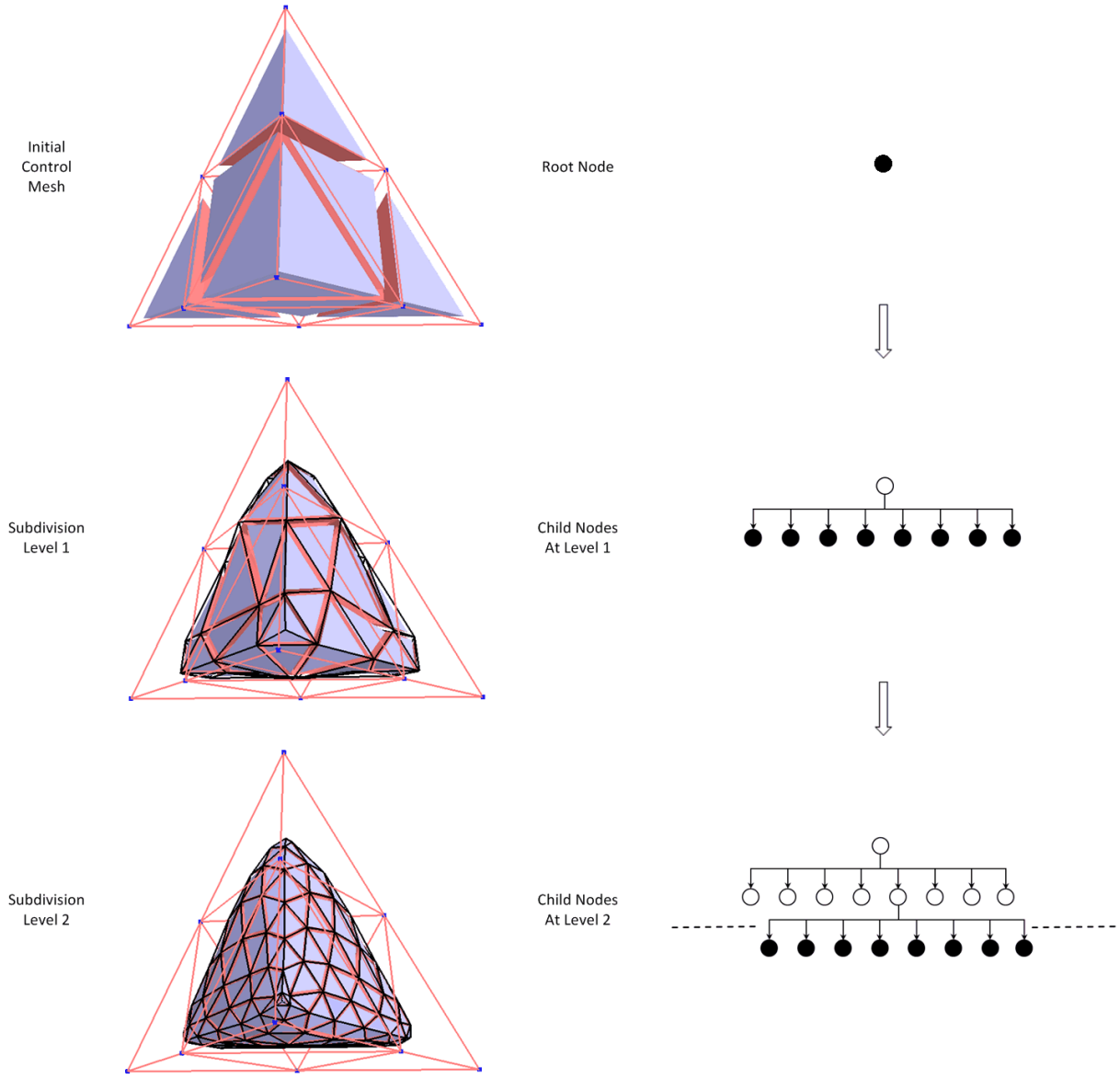

Figure 3.14: The hierarchy of a tetrahedral mesh at multiple subdivision levels generated through iteratively applying the solid Loop subdivision.

3.2.3 Linear Transformation-based Tetrahedron Shape Measure

The solid Loop subdivision-based refinements presented previously allows to uniformly refine a tetrahedral mesh, which involves generating new tetrahedral meshes at multiple subdivision levels. Since the subdivision-based meshes is served for numerical simulations, it is important to make sure that the meshes generated during multiple subdivision iterations are not degenerated.

For this reason, the quality of the subdivided tetrahedrons is evaluated using an effective quantitative means: a linear transformation-based tetrahedron shape measure, which is discussed thoroughly in the work of Dompierre et al. (Dompierre, Vallet et al. 2005). This shape ratio measure of tetrahedron elements η is detailed in Equation 3.5:

$$\eta = \frac{12\sqrt[3]{9V_k^2}}{\sum_{1\le i<j\le 4} L_{ij}^2}, \tag{3.5}$$

where V_k represents the volume of a tetrahedron and L_{ij} represents the tetrahedron edge lengths.

A tetrahedron is formed by four vertices $\{v_1, v_2, v_3, v_4\}$ and six edges, whose corresponding edge lengths L_{ij} are expressed in Equation 3.6:

$$L_{ij} = \|v_j - v_i\|. \tag{3.6}$$

The tetrahedron volume V_k can be calculated through its edge lengths (see Equation 3.7 (Pólya 1964)):

$$\begin{aligned} 144V_k^2 = {} & 4L_{12}^2L_{13}^2L_{14}^2 + (L_{13}^2 + L_{14}^2 - L_{34}^2)(L_{14}^2 + L_{12}^2 - L_{24}^2)(L_{12}^2 + L_{13}^2 - L_{23}^2) \\ & - L_{12}^2(L_{13}^2 + L_{14}^2 - L_{34}^2) - L_{13}^2(L_{14}^2 + L_{12}^2 - L_{24}^2) - L_{14}^2(L_{12}^2 + L_{13}^2 - L_{23}^2). \end{aligned} \tag{3.7}$$

The *simplex shape measure* η is a continuous function, which is invariant under translation, rotation, reflection and valid uniform scaling of the tetrahedron simplex. The tetrahedron shape measure is valid only under the condition of $\eta \in [0,1]$. If η equals 1, the evaluated simplex is regular, otherwise, if $\eta = 0$, the evaluated shape is degenerate. In the work of Dompierre et al. (Dompierre, Vallet et al. 2005), the authors presented various tetrahedron shape measures and summarized the η tetrahedron measure as the best choice due to the reasons, such as it is less expensive to calculate and its computation is numerically stable.

Since the subdivision-based refinement operation includes two sub-operators: topological splits and geometrical smoothing, the η tetrahedron shape measure is applied both on the split tetrahedral meshes without smoothing and on the subdivided tetrahedral meshes with smoothing

at different subdivision levels. The concrete η measure includes: average η, minimum η and maximum η. Our measure begins with a regular tetrahedron with edge-length 2, whose four corner vertices' Cartesian coordinates are listed in Equation 3.8 (Wikipedia.org):

$$\begin{aligned} v_1 &= \left(1, 0, -1/\sqrt{2}\right), \\ v_2 &= \left(-1, 0, -1/\sqrt{2}\right), \\ v_3 &= \left(0, 1, 1/\sqrt{2}\right), \\ v_4 &= \left(0, -1, 1/\sqrt{2}\right). \end{aligned} \tag{3.8}$$

Firstly, full 1-to-8 topological splits are performed on the regular tetrahedron. This split does not involve smoothing. The same tetrahedron is split four times, which generates four tetrahedral meshes with different LODs (see Figure 3.15). The η evaluation results of the split regular tetrahedron meshes is listed in the Table 3-6.

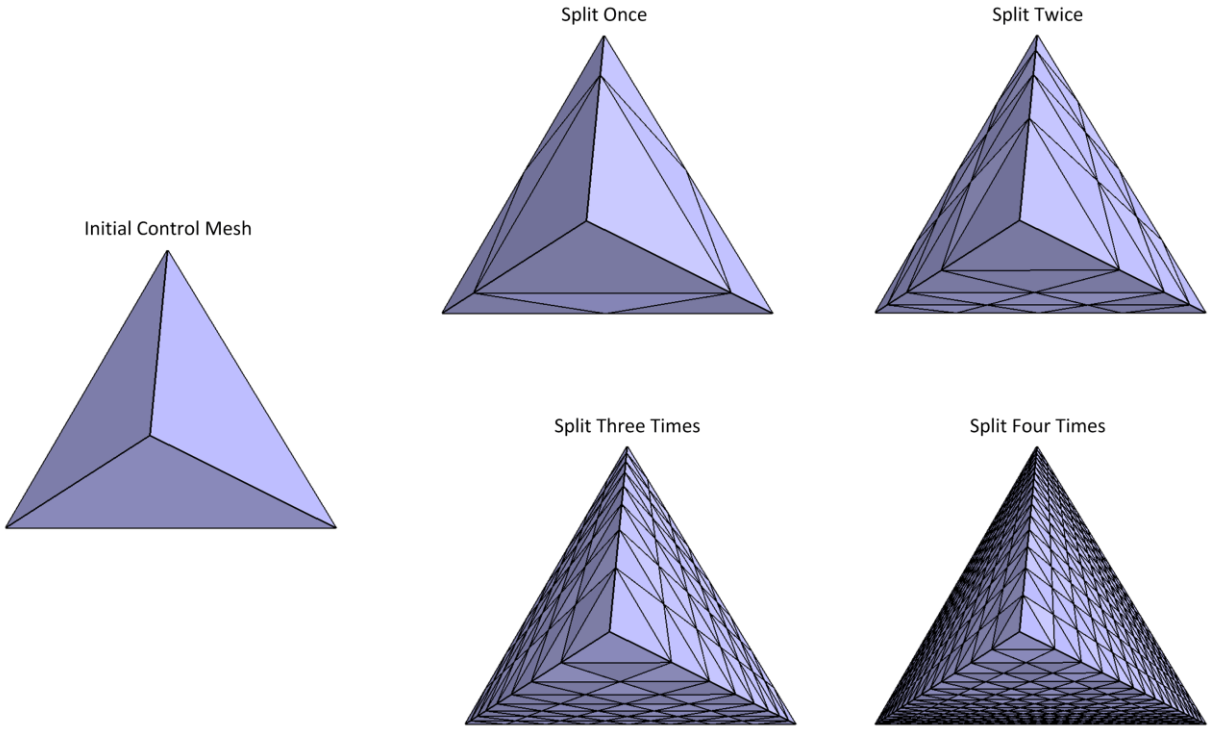

Figure 3.15: 1-to-8 topological splits applied on a regular tetrahedron mesh from once to four times.

Table 3-6: The corresponding η evaluations on the four split regular tetrahedron meshes.

Evaluation \ Split Level	Initial	Level 1	Level 2	Level 3	Level 4
minimum η	1.000000	0.857143	0.857143	0.857143	0.857143
maximum η	1.000000	1.000000	1.000000	1.000000	1.000000
Average η	1.000000	0.928571	0.910714	0.906250	0.905134

From Table 3-6, we can observe that from split level 1 to split level 4, the maximum η is maintained at 1, the minimum η is once turned into 0.857143 at the first subdivision level and then is stabilized in the same value 0.857143, and the average η ranges between 0.928571 and 0.905134, which implies that as the split levels increase, even though the average shape ratio value decreases, the corresponding mesh quality is generally stabilized.

Table 3-7: The corresponding η evaluations on the four split cube tetrahedral meshes.

Evaluation \ Subdivision Level	Initial	Level 1	Level 2	Level 3	Level 4
minimum η	0.687230	0.503968	0.503968	0.503968	0.503968
maximum η	1.000000	1.000000	1.000000	1.000000	1.000000
Average η	0.810330	0.781751	0.765538	0.753754	0.745175

The evaluation above is then applied on a cube example. This cube example is usually used for demonstrating the geometrical smoothing effect, since its initial control mesh is contoured with a cube form whereas after smoothing, it converges toward an interior approximation sphere. Here we verify only if the 1-to-8 topological split degenerates the split mesh quality on the cube example. The four split cube tetrahedral meshes are illustrated in Figure 3.16.

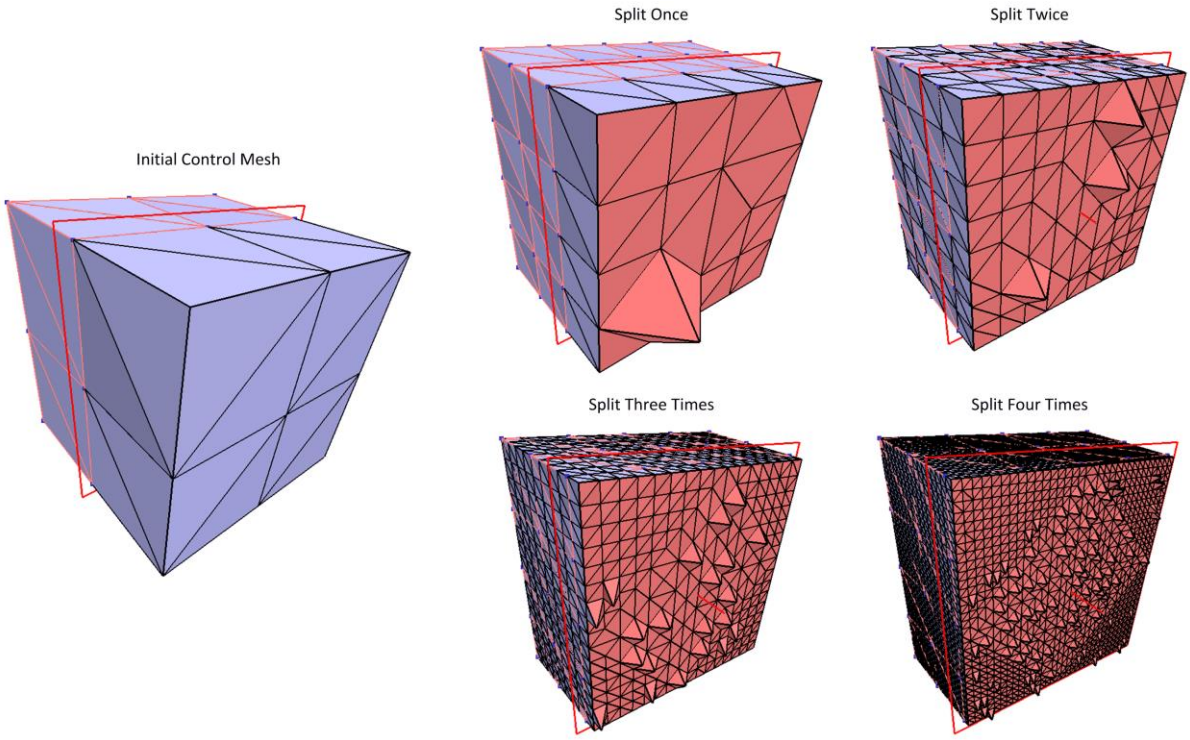

Figure 3.16: 1-to-8 topological splits applied on a cube tetrahedral mesh from once to four times.

We list the corresponding η values measured for different split levels in Table 3-7. From this table, we can observe that after being split once, the minimum η values are maintained at the same value 0.503968, which do not decrease as split levels increase. The maximum η values equal always 1. The average η value starts at 0.810330 and then it declines slowly and stably from 0.781751 to 0.745175 as the split levels transit from level 1 to level 4.

According to the evaluation results of these two examples, we can conclude that the 1-to-8 topological split does not degenerate tetrahedral mesh quality. Besides, in response to the fact that the minimum η values are stabilized at the same values after the first splits, we consider it logical in the reason that once the split pattern is set up from the first time, the subsequent splits follow the same pattern. That is to say, the 1-to-8 topological split pattern in our work can be used for splitting the tetrahedral meshes with favorable η values for numerical simulations.

Subsequently, we evaluate the mesh quality of subdivision-based refined tetrahedral meshes, which are generated by applying the solid Loop subdivision four times on the two examples above: the regular tetrahedron mesh (see Figure 3.17) and the cube tetrahedral mesh (see Figure 3.18).

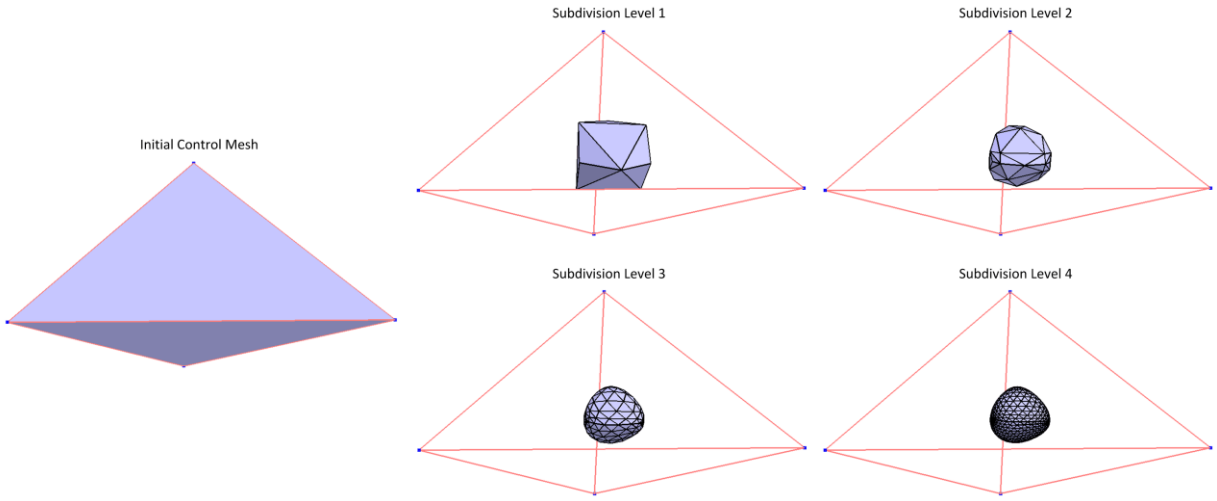

Figure 3.17: Solid Loop subdivision applied on a regular tetrahedron from once to four times.

We list the corresponding η evaluations on the subdivided regular tetrahedron results in Table 3-8 and the subdivided cube tetrahedral mesh results in Table 3-9. In order to display the η trend over subdivision levels, we use line charts (see Figure 3.19 and Figure 3.20) to present the results from Table 3-8 and Table 3-9.

According to the evaluation results on the refined subdivision meshes at different subdivision levels, we can observe that during the first subdivision, the minimum η and maximum η values rapidly drop and then during the subsequent subdivision iterations, the minimum η values approach towards 0 whereas the maximum η values approach toward 1. The average η values are stabilized as the corresponding subdivision levels increase.

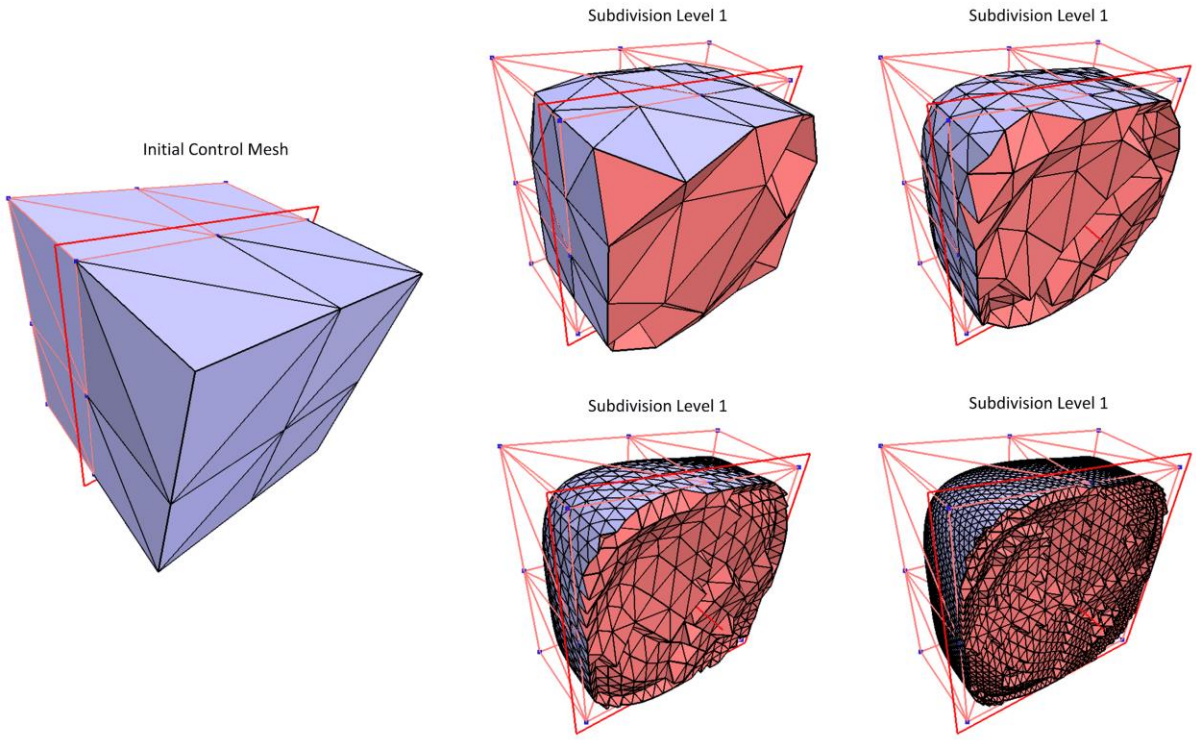

Figure 3.18: Solid Loop subdivision applied on the cube tetrahedral mesh from once to four times.

Even though these evaluation results, particularly the minimum η values, conclusively demonstrate that mesh quality of tetrahedral subdivision meshes generated from uniform subdivision iterations are degenerated, we must emphasize that these results actually correspond to the refinement experiences in practice. As what we can observe from Figure 3.17 and Figure 3.18, the boundary surfaces of the two tetrahedral mesh examples are severely shrunk towards the volume interior during the first subdivision iteration in comparison with the refined boundary surfaces generated during the subsequent subdivision iterations. The boundary surface shrinkage consequently causes the shapes of the tetrahedrons on boundary surfaces becoming compressed and flat, whereas the flatness of boundary tetrahedrons evolved from the transitions between two different subdivision iteration are inconstant. In some way, this fact could explain why the maximum η values drop sharply during the first subdivision iterations and then rebound back toward 1 during the subsequent subdivision iterations.

Table 3-8: The corresponding η evaluations on the four subdivided tetrahedron meshes.

Subdivision / Evaluation / Level	Initial	Level 1	Level 2	Level 3	Level 4
minimum η	1.000000	0.577237	0.235775	0.000000	0.037204
maximum η	1.000000	0.857143	0.988474	0.997444	0.999179
Average η	1.000000	0.717190	0.659548	0.647117	0.652620

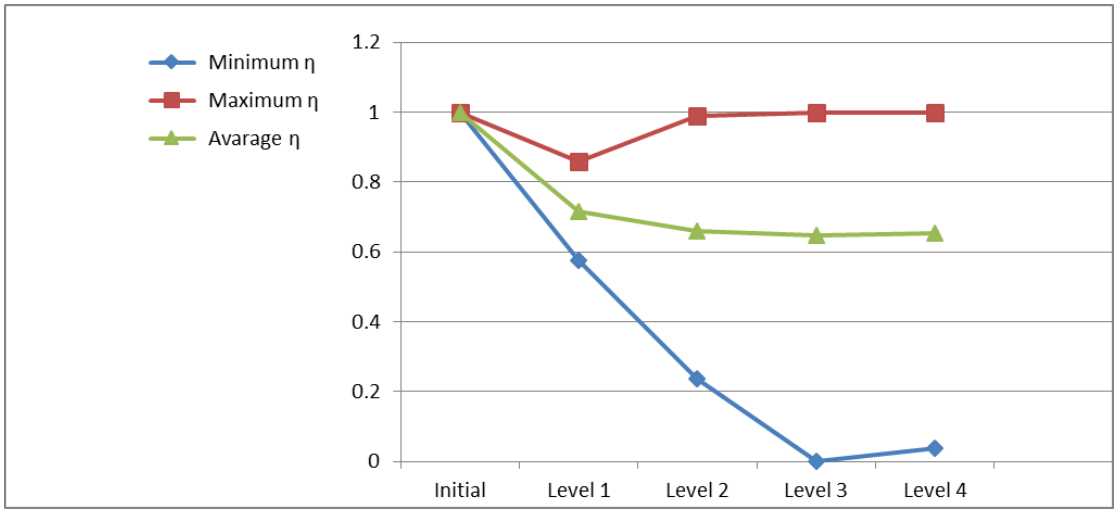

Figure 3.19: η evaluations on the subdivided regular tetrahedron meshes.

Table 3-9: The corresponding η evaluations on the four subdivided cube tetrahedral meshes.

Subdivision / Evaluation / Level	Initial	Level 1	Level 2	Level 3	Level 4
minimum η	0.687230	0.372093	0.118812	0.059057	0.022791
maximum η	1.000000	0.966418	0.970822	0.998480	0.999082
Average η	0.810330	0.718071	0.713621	0.726968	0.742572

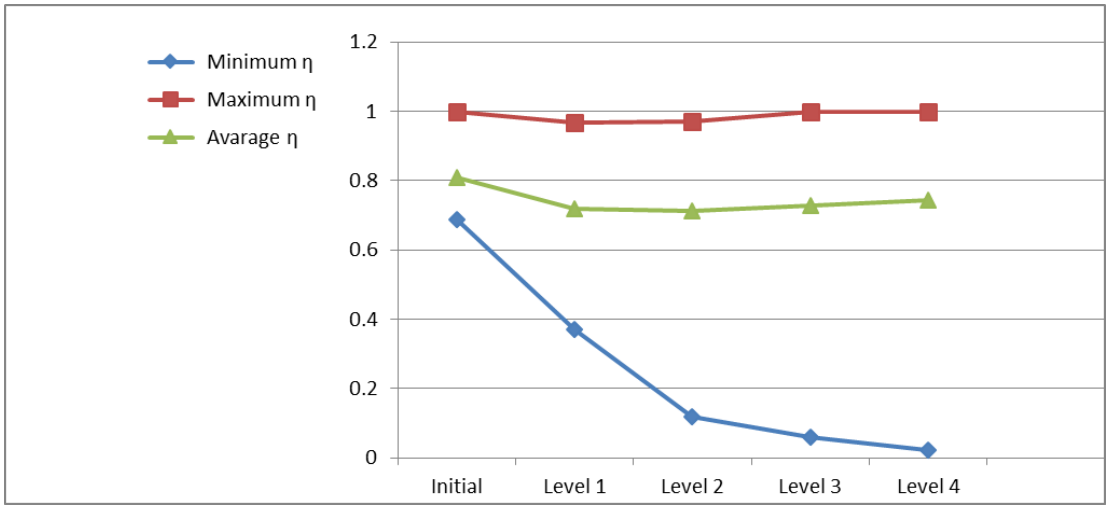

Figure 3.20: η evaluations on the subdivided cube tetrahedral meshes.

In conclusion, the geometric smoothing on boundary surfaces increases the possibility of generating tetrahedrons with undesirable shapes. Besides, uniform subdivisions are not favorable in isolating these tetrahedrons as subdivision iterations continue. All these factors consequently result in the generated subdivision tetrahedral meshes to degenerate after several refinement steps.

3.2.4 Proposition of Adaptive Subdivision

The disadvantages of using the traditional subdivision-based refinement method are summarized as follows: 1) It is based on uniform subdivisions, which lacks flexibility. This factor does not facilitate fulfilling practical requirements such as controlling the mesh density of local volumes or isolating the undesirable mesh parts to lessen the possibilities of tetrahedral mesh degeneracy. 2) It is an iterative approximation method, which is to approximately approach the true geometry through subdivision iterations. The factor puts the system in the risk of heavily consuming storage and computation resources. This risk is aggravated when we use an eight-child tree structure to keep multiple subdivision-based tetrahedral meshes at different subdivision levels, where subdivision entities are duplicated 8 times through each subdivision iteration.

The idea of integrating adaptive subdivision into our refinement works emerges as a way to avoid the drawbacks of uniform subdivisions. Adaptive subdivision consists in refining the mesh parts as needed instead of completely subdividing everywhere. However, integrating adaptivity

support into solid subdivision brings up one problem: how to stitch together various mesh parts with different resolution details?

This problem is challenging due to two aspects: 1) in order to directly access the required mesh part at the specific subdivision level, it is necessary to facilitate navigation among the different subdivision levels. Consequently, the connectivity information in the eight-child tree structure guiding the access from the higher subdivision levels to the lower subdivision levels or inverse (parent inheritance and children retrospect) must be kept. Evidently, this kind of information is even more cumbersome and complicated, especially on volume meshes; 2) to merge various volumetric mesh parts with different subdivision resolutions involves dealing with tetrahedrons with poor angles, such as needles, wedges or slivers, which require another complicated post-optimization process (Shewchuk 1998, Volkov and Ling 2003). In conclusion, these two aspects cannot be solved by only recurring to the traditional subdivision-based refinement method, which combines 1-to-8 topological splits and uniform subdivision-based refinements.

3.3 Integration of Adaptivity Support into Solid Subdivision

To improve the precision and the efficiency in subdivision-based representations, two essential elements are merged into the traditional subdivision-based refinement method: 1) the subdivision surface parameterization techniques; 2) adaptive refinements.

The subdivision surface parameterization techniques are introduced into the proposed solid subdivision scheme, which permits the limit position of any boundary surface vertex on 3D tetrahedral meshes to be calculated. This introduction eliminates the redundant approximation iterations by directly projecting the refined mesh to the target boundary surface representing the true geometry.

Adaptive refinements combine adaptive tetrahedron splits with refinement criteria, which permit to adaptively split the tetrahedrons while conforming to specific refinement criteria. Adaptive refinement replaces the mandatory 1-to-8 full tetrahedron split of uniform subdivision. This replacement contributes to reasonably controlled LODs according to actual geometric modification requirements from real applications.

To merge the two elements above, we develop a single-level refinement method. This method insures that the boundary surface vertices meeting refinement criteria are accurately projected to their limit positions. This eventually generates a multi-resolution subdivision-based tetrahedral meshes composed of various volume parts at different resolutions.

CHAPTER 4
ADAPTIVE SUBDIVISION-BASED REFINEMENTS

Adaptive subdivision emerges as a requirement to improve precision and efficiency when implementing the solid Loop-based subdivision scheme into real applications. To illustrate this new subdivision-based refinement method, implementations in three types of subdivision-based representations: curve, triangular surface and tetrahedral meshes are presented.

In this chapter, first we clarify the difference between the traditional refinement method and the single-level method that we have developed by applying them on the same examples. Then we focus on presenting concrete adaptive refinement algorithms particularly for curve and surface examples. At the end of this chapter, we present adaptive refinements on tetrahedral meshes.

4.1 Exact Evaluation on Subdivision Curves

In Section 3.1, we demonstrated the gradual approximation procedure of approaching a target geometry: an approximate circular shape from a square curve through traditional subdivision iterations (see Figure 3.13). In this curve example, the control vertices at the former subdivision levels are updated to their new positions at the higher subdivision levels during subdivision iterations.

4.1.1 Traditional Subdivision-based Curve Refinements

The existing vertex at the new subdivision level p_{2i}^{j+1} is influenced mainly by its former position p_i^j and partly by the two direct neighbouring vertices p_{i-1}^j and p_{i+1}^j at the previous subdivision level (see Equation 4.1).

$$p_{2i}^{j+1} = (p_{i-1}^j + 6p_i^j + p_{i+1}^j)/8. \tag{4.1}$$

The edge splits result in new vertices being inserted in the middle of each line segment at the former subdivision levels. These newly inserted vertices p_{2i+1}^{j+1} are defined only by the two endpoints of the split edge $\left(p_i^j, p_{i+1}^j\right)$ (see Equation 4.2).

$$p_{2i+1}^{j+1} = (p_i^j + p_{i+1}^j)/2. \tag{4.2}$$

Equation 4.1 and Equation 4.2 are respectively the Loop subdivision rules for the even vertices and the odd vertices on boundaries or creases of Loop subdivision surfaces (see Equation 2.14 and Equation 2.13). These two equations are merged into a linear operator: the subdivision mask S^T (see Equation 4.3), which indicates the influence of the control vertices at the j^{th} subdivision level to the ones at the $(j+1)^{th}$ subdivision level on subdivision curve meshes.

$$S^T = \frac{1}{8}[1 \;\; 4 \;\; 6 \;\; 4 \;\; 1]. \tag{4.3}$$

In a traditional subdivision-based curve refinement method, the refinement operator is replaced by the subdivision mask above.

4.1.2 Loop Subdivision Curve Limits

After applying the subdivision mask an infinite number of times, the square curve mesh converges to a smooth circular curve L. In De Boor's B-splines book (De Boor 2001), the author demonstrates that this subdivision curve limit is in fact a cubic B-spline curve. In Persson et al.'s work of using Loop subdivision surfaces for surrogate geometry, the authors presented one evaluation subdivision mask (see Equation 4.4) (Persson, Aftosmis et al. 2006):

$$p_i^\infty = (p_{i-1}^0 + 4p_i^0 + p_{i+1}^0)/6. \tag{4.4}$$

The limit position of any control vertex p_i^∞ can be precisely computed by directly applying this evaluation subdivision mask once on its own position p_i^0 together with its direct neighboring control vertices p_{i-1}^0 and p_{i+1}^0 on the initial control curve mesh M^0. For example, the limit position p_0^∞ in the square curve example can be calculated by applying Equation 4.4 on the three initial control vertices $\{p_3^0, p_0^0, p_1^0\}$ (see Figure 4.1(a)).

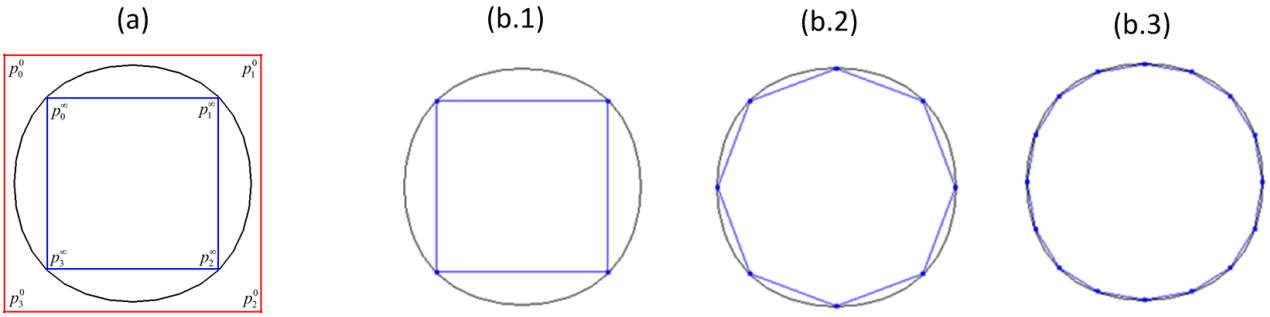

Figure 4.1: Uniformly bisecting the square curve edges and projecting the existing and the newly inserted vertices onto their limit positions of the subdivision limit.

To formulate a direct mapping between an initial control mesh M^0 and its subdivision limit L, we use the evaluation subdivision matrix presented in the same article of Persson et al. (Persson, Aftosmis et al. 2006). Taking the same square curve as an initial control mesh example, we introduce a parameter value u to define the exact position where the evaluated vertex lies on an initial line segment (p_i^0, p_{i+1}^0) and $0 \le u \le 1$. Then we calculate the limit position of any arbitrary vertex $p_{(i,i+1)}^0(u)$ by using the following evaluation matrix (see Equation 4.5):

$$p_{(i,i+1)}^{\infty}(u) = \frac{1}{6}\begin{bmatrix} u^3 & u^2 & u^1 & 1 \end{bmatrix}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}\begin{bmatrix} p_{i-1}^0 \\ p_i^0 \\ p_{i+1}^0 \\ p_{i+2}^0 \end{bmatrix}. \tag{4.5}$$

The analytic information on the corresponding vertex, such as the curvature and tangent can be directly obtained by deriving the evaluation matrix above. For example, $p_{(i,i+1)}^{\infty}{}'(u)$ is calculated by deriving the vector $\begin{bmatrix} u^3 & u^2 & u^1 & 1 \end{bmatrix}$ once (see Equation 4.6).

$$p^{\infty}_{(j,j+1)}{}'(u) = \frac{1}{6}\begin{bmatrix} 3u^2 & 2u^1 & u^0 & 0 \end{bmatrix}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}\begin{bmatrix} p^0_{j-1} \\ p^0_j \\ p^0_{j+1} \\ p^0_{j+2} \end{bmatrix}. \quad (4.6)$$

It is important to note that initial control meshes must be closed curves. It is not feasible to evaluate the endpoints of open curves by using the equations above, since any curve endpoint has only one direct neighbouring vertex. According to our study, there are no other pertinent researches available. In brief, this subdivision evaluation technique can be used for projecting any arbitrary position from initial control meshes with closed forms onto the limit position of their corresponding subdivision limits. For example, we uniformly bisect the curve edges of the same square curve mesh, and then project all the existing vertices and the newly inserted vertices onto their limits (see the examples (b.2) and (b.3) in Figure 4.1).

Since subdivision limits represent interesting properties in surface continuity and analytic geometry, we use subdivision-based representations to represent the true geometry of the studied models. In the case of curve refinements, a subdivision-based curve mesh consists of vertices, which are in fact the limit subdivision vertices on the subdivision curve limit. Because of the lack of flexibility of uniform refinements in controlling concrete level of details as required, subdivision curve meshes are adaptively refined, while the newly inserted edge vertices are directly projected onto their limit positions.

4.1.3 Adaptive Curve Refinement Algorithms

The algorithms for supporting adaptive curve refinements are detailed as the two operations below: 1) inserting a new vertex between the selected edge (p_i, p_j) on the refined mesh; 2) applying the refinement criterion.

The first part is presented in the pseudo codes of $SplitEdge(p_i, p_j)$ (see Table 4.1). In this function, the ratio value is defined as ‘0.5’ by default, since the new edge vertex is inserted in the middle of the selected edge. Certainly this ratio could vary according to actual users’ specifications. The sub-function $LocateInitialEdgeSegment()$ is used to locate the corresponding

edge segment of p_i and p_j on the initial control mesh. In the definition of each vertex, it contains the information from which initial edge segment comes. The sub-function *EvaluateArbitraryVertex*() is integrated with Equation 4.5 for exactly evaluating the inserted vertex. This sub-function also serves as indexing the four neighbouring control vertices on the initial curve involved to determine the limit position of the newly inserted vertex on the refined curve.

Table 4-1: The pseudo codes for inserting a new vertex between the selected edge (p_i, p_j) on the refined mesh

```
SplitEdge(v_i, v_j)
{
        //Define the value u to locate the inserted position
        ratio = 0.5;
        u = (1.0 - ratio) *pi.u + ratio * pj.u;
        //Locate the original edge on the initial control mesh
        Index_OriginalEdge = LocateInitialEdgeSegment(pi, pj);
        //Calculate the limit positions of the newly inserted vertex
        LimitofInsertedVertex = EvaluateArbitraryVertex(u, Index_OriginalEdge);
        Insert LimitofInsertedVertex between pi and pj;
}
```

The second part is presented in the function *SplitCurveAdaptive*() (see Table 4.2). This function involves the definition of refinement criteria. Usually, these criteria are specified according to practical requirements. Here, we use a measure based on curvature (see Equation 4.7) (wikipedia.org) to guide adaptive curve refinements:

$$h > c / k, \tag{4.7}$$

where h represents the length of the queried edge segment (p_i, p_{i+1}), c represents a constant default value and k represents the second derivative of the current segment. Here we use the sub-function *EvalCurvature*2() to calculate the value k by evaluating the middle point of the selected segment $p^{\infty}_{(i,i+1)}{}''(u)$.

Table 4-2: The pseudo codes for adaptively refining curves

```
SplitCurveAdaptive()
{
        const float c = 0.1f;
        do
        {
                For (Each Vertex i on the Refined Curve)
                {
                        p1 = Points[i];
                        p2 = Points[i+1];
                        SegmentLength = (p1 – p2).norm();
                        //Evaluate the curvature at the center of this segment
                        p = (p1.v + p2.v) * 0.5f;
                        k = EvalCurvature2(p, p1.originalEdge).norm();
                        if(LengthofEdgeSegment_h > (c/ k))
                        {
                                SplitEdge (p1, p2);
                        }
                }
        }while(SomethingGotSplit)
}
```

In summary, the adaptive curve refinement algorithms in our work refine the curves until the refinement criteria are met. Here the refinement operation is to bisect curve edges and project the newly inserted middle edge vertex into its limit position.

4.1.4 Results of Adaptive Curve Refinements

In this section, we demonstrate the results of applying adaptive refinements on the same curve mesh by varying the value c defined in Equation 4.7 as 0.5, 0.2, 0.05 and 0.01. The corresponding refinement results are illustrated in Figure 4.2, where the initial square curve mesh is displayed in red lines, its subdivision limit is displayed as an approximate circular curve in black, the refined mesh is displayed as the curve in blue, and the vertices' derivatives of the refined mesh are displayed in cyan lines.

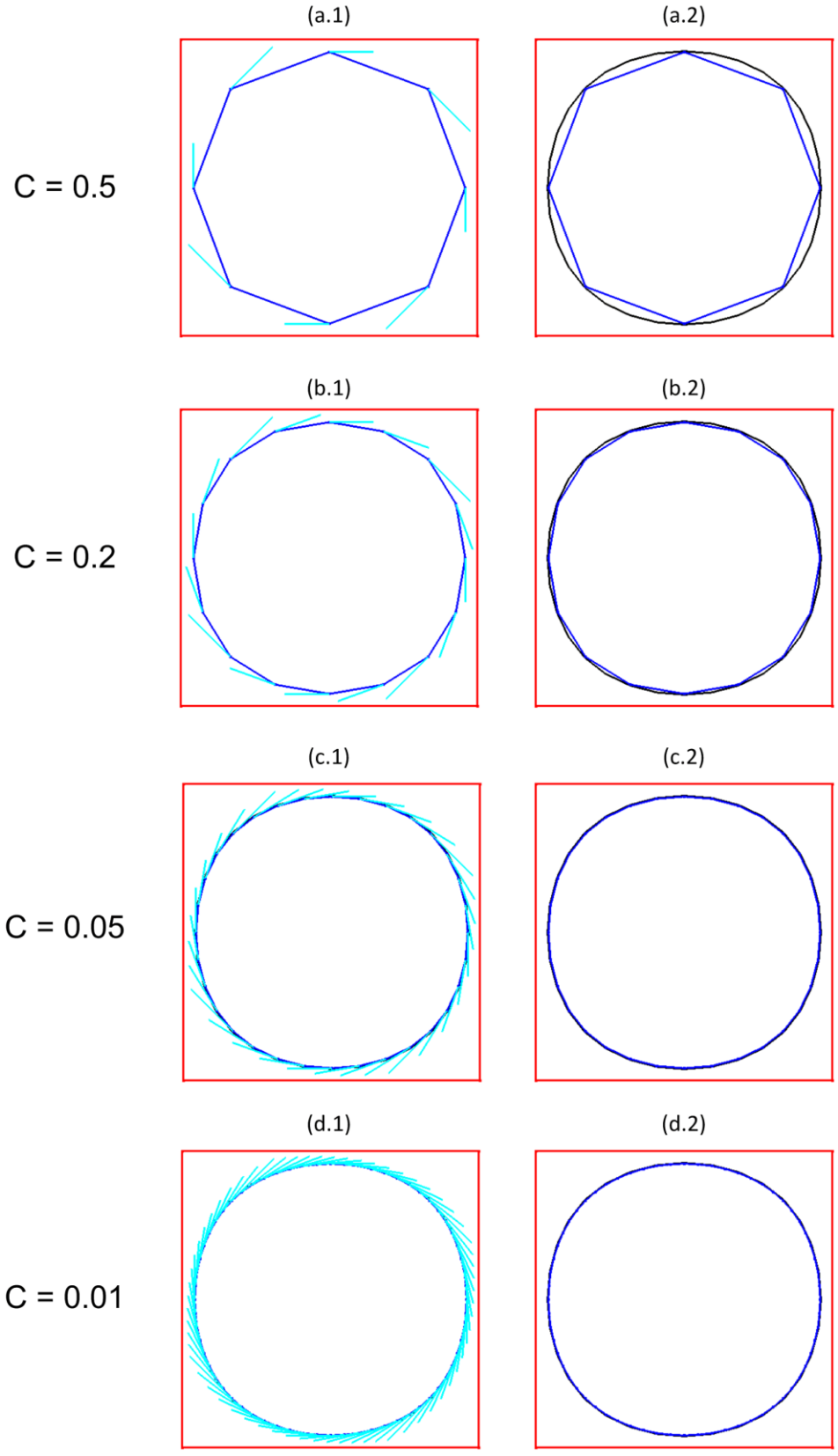

Figure 4.2: Adaptively refining the square curve mesh by varying the value c.

From Figure 4.2, it is observed that as the value c is decreased, the difference between the refined mesh and the subdivision limit also decreases.

Here, we present another curve example related to simulating a RAE2822 airfoil. In this presentation, three curves are involved (see Figure 4.3 (a)): 1) Curve_1 - limit curve (in black): this curve is the target curve. We suppose that this curve is exactly the geometry presentation of a RAE2822 transonic airfoil. The coordinates of the RAE2822 transonic airfoil come from the UIUC airfoil data site (UIUC). 2) Curve_2 - Subdivision Friendly Curve (in red): this curve is obtained by using the modeling software Rhinoceros. This curve is treated as the initial control mesh of Curve_1. As discussed, the inverse subdivision is beyond our research scope. Here, the accuracy of Curve_2 is approximated. 3) Curve_3 - The start curve (in blue): this curve consists of vertices, whose positions are on the subdivision limit Curve_2. The density of Curve_3 is controlled by refinement criteria. In order to better visualize the curve details, we zoom in on the airfoil leading edge in Figure 4.3(b).

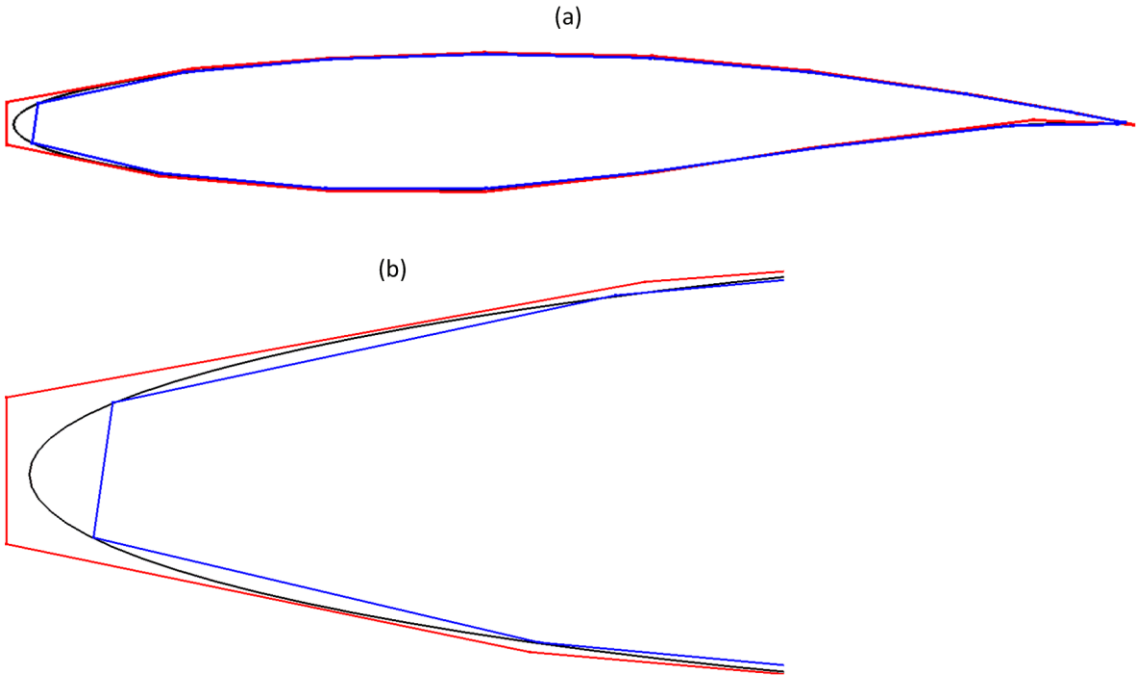

Figure 4.3: The illustration of three curves: Curve_1, Curve_2, Curve_3.

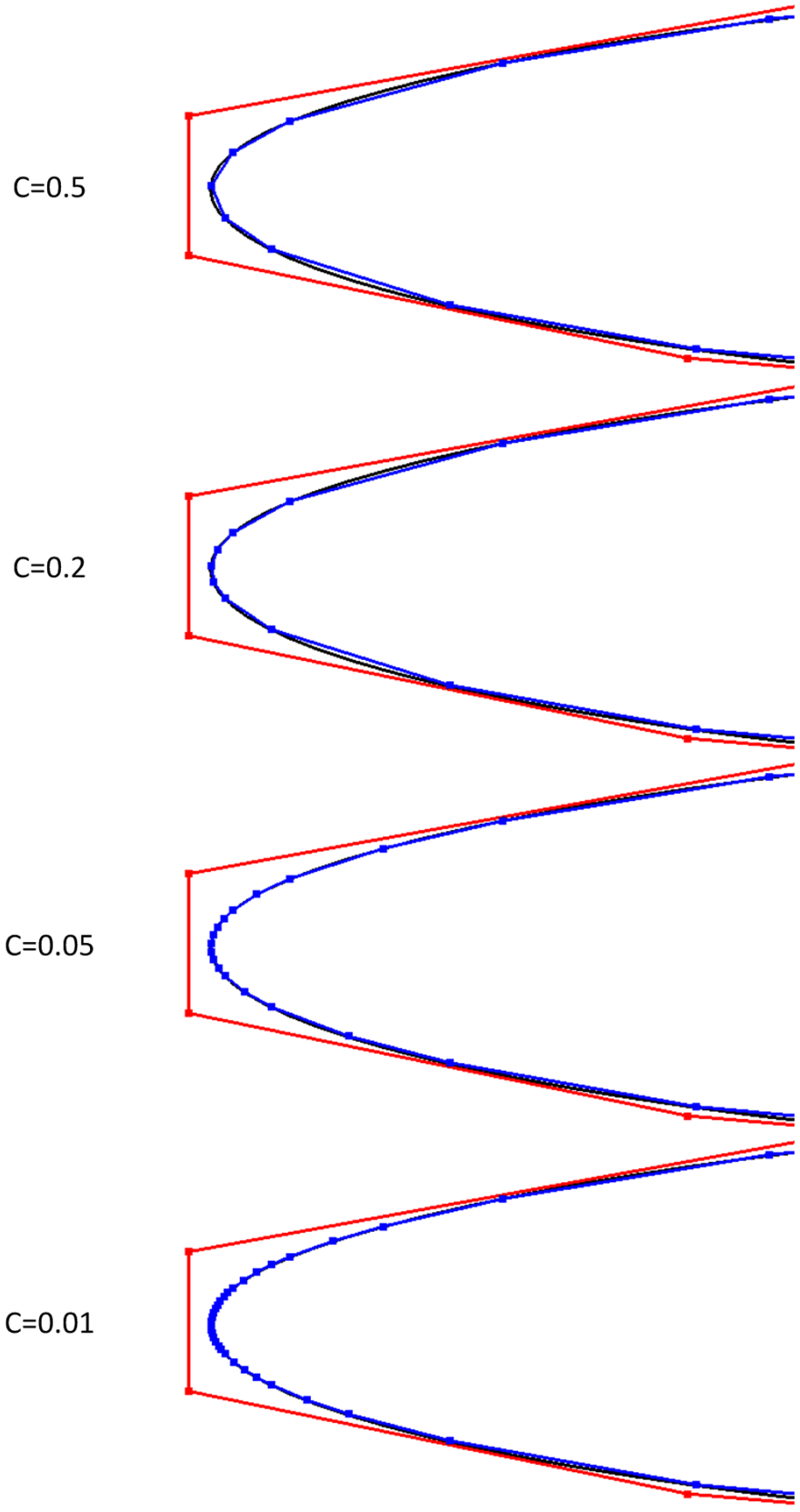

Figure 4.4: Adaptively refining the RAE2822 airfoil curve mesh by varying the value c.

The LODs of the refined RAE2822 airfoil (Curve_3 in blue) is gradually and adaptively increased (see Figure 4.4) according to the same criterion used for refining the square curve example (see Figure 4.2). From the results of the two examples above, we can conclude that our adaptive curve refinement method can be successfully used for adaptively refining curve meshes. The adaptively refined curve meshes are exactly on the subdivision limit, whose accuracy can be controlled according to the concrete refinement criteria.

4.1.5 Single-level Refinement Method

In this section, we present the single-level refinement method, whose basic concepts can be equally used for refining surfaces and volume meshes. We develop this method for facilitating the integration between subdivision technique and adaptive refinements. The single-level refinement method distinguishes itself from the multi-level non-uniform refinement method. In Figure 4.5, we illustrate the two curve meshes refined respectively by using these two refinement methods.

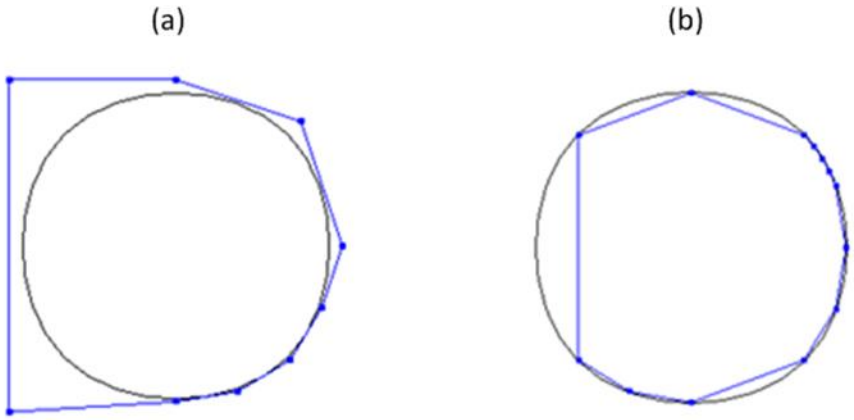

Figure 4.5: The comparison of two different refinement methods: a) the multi-level non-uniform refinement method; b) the single-level refinement method.

The whole curve mesh (illustrated in Figure 4.5(a)) consists of mesh parts at different levels of details. Each mesh part is obtained by individually applying traditional subdivisions on the basis of local iteration requirements. That is to say, globally this mesh is an approximation of the target geometry. The approximation result is composed of multiple local mesh parts, which are approximated by approaching them as closely as possible with the corresponding parts in the

target geometry. More specifically each local approximation involves refining the same initial control mesh part multiple times and then generating multiple subdivision partial meshes.

To support the data structure used in the multi-level non-uniform refinement method could require heavy storage, since all the hierarchical connectivity information (parent inheritance and children retrospect) for facilitating the navigation among multiple subdivision levels together with the identification information regarding the corresponding mesh parts must be kept. The storage requirements could be much more cumbersome, particularly for meshes with massive geometric data and complex topology. Besides, to stitch together the different surface or volume mesh parts with various LODs could be challenging, since cracks or T-vertices frequently emerge as this refinement method is used for refining surface or volume meshes (Volkov and Ling 2003).

Differently, the two issues above are avoided by using our single-level refinement method (see Figure 4.5(b)). In this single-level method, the subdivision parameterization technique is introduced so that the limit position of any arbitrary position on the initial control mesh can be obtained. This parameterization is a one-to-one mapping, which only involves the studied positions on the initial control mesh and their corresponding mapped positions on the subdivision limit. In other words, each refinement only updates the density of the selected mesh parts, and even after many refinements, there are only two meshes that are kept in the system: the initial control mesh and the refined mesh, whose detailed information are respectively stored in Mesh_1 and Mesh_2.

More specifically, Mesh_1 is a mesh consisting of the initial control vertices. Mesh_1 is constantly unchanged and only supplies information for Mesh_2. Mesh_1 is defined right from the beginning of subdivision-based refinements. Mesh_2 is a mesh consisting of the vertices, whose positions are on the subdivision limit. When adaptive subdivision is applied, new vertices are adaptively added on the refined mesh. These newly added vertices are inserted into Mesh_2. Thus Mesh_2 is updated in real-time as refinements continue. Thus in the whole refinement process, there are only Mesh_1 and Mesh_2. This fact is different from the multi-level method, which requires a multi-level treelike structure to keep the hierarchical information of the mesh parts generated from different subdivision levels.

Moreover, between Mesh_1 and Mesh_2 in the single-level method, the subdivision-based refinement operation plays the role of projecting. This refinement operation is specified by

parameterization techniques in practice. According to the information in Mesh_1, we can obtain any limit position on the subdivision limit. Our effort is to project the limit positions meeting the defined refinement criteria and insert them into Mesh_2. That is to say that the resolution of Mesh_2 is determined by the specifications of refinement criteria.

About the stitching problem when adaptively refining surface or volume meshes, in our work we introduce different topological split patterns. In the following two sections, we are going to demonstrate that these split patterns can be naturally merged with the projection and insertion process above for eliminating the stitching problem.

4.2 Adaptive Refinements on Triangular Surfaces

When the adaptive refinements are extended from curves to surface or volume meshes, the basic elements for split vary from edge segments to triangular faces or tetrahedrons. This reality indicates that the involved topologies are to be treated differently.

4.2.1 Refinements on a Square Surface Example

In our adaptive refinement work, triangular surfaces are treated in two parts: the boundary and the interior. For one triangular surface mesh, its boundary is a curve connected with a collection of edge segments and its interior is a surface embedded with triangular faces. The adaptive triangular surface refinement can be summarized as: 1) the boundary curve is treated as one subdivision curve, whose edge segments are adaptively bisected. And then the newly inserted edge vertices are projected to their boundary curve limits. 2) At the same time, the interior triangular faces are split into 4 or 2 smaller triangular faces according to the tag information on the corresponding facet edges.

We present the procedure for adaptively refining a square surface example together with detailing the respective subdivision operations on the different surface parts. The square surface mesh M^0 is constructed with four initial control vertices $\left\{p_0^0, p_1^0, p_2^0, p_3^0\right\}$. It consists of a boundary square curve (displayed in red) and an interior surface separated into two triangles by an edge (displayed in green) (see Figure 4.6(a)). The boundary square curve is connected with 4 edge segments $\left(p_0^0, p_1^0\right),\left(p_1^0, p_2^0\right),\left(p_2^0, p_3^0\right)$ and $\left(p_3^0, p_0^0\right)$. The interior includes two triangular

faces $\Delta p_0^0 p_1^0 p_3^0$ and $\Delta p_1^0 p_2^0 p_3^0$. The limit subdivision boundary mesh L is represented by an approximate closed circular curve inside the initial square surface mesh.

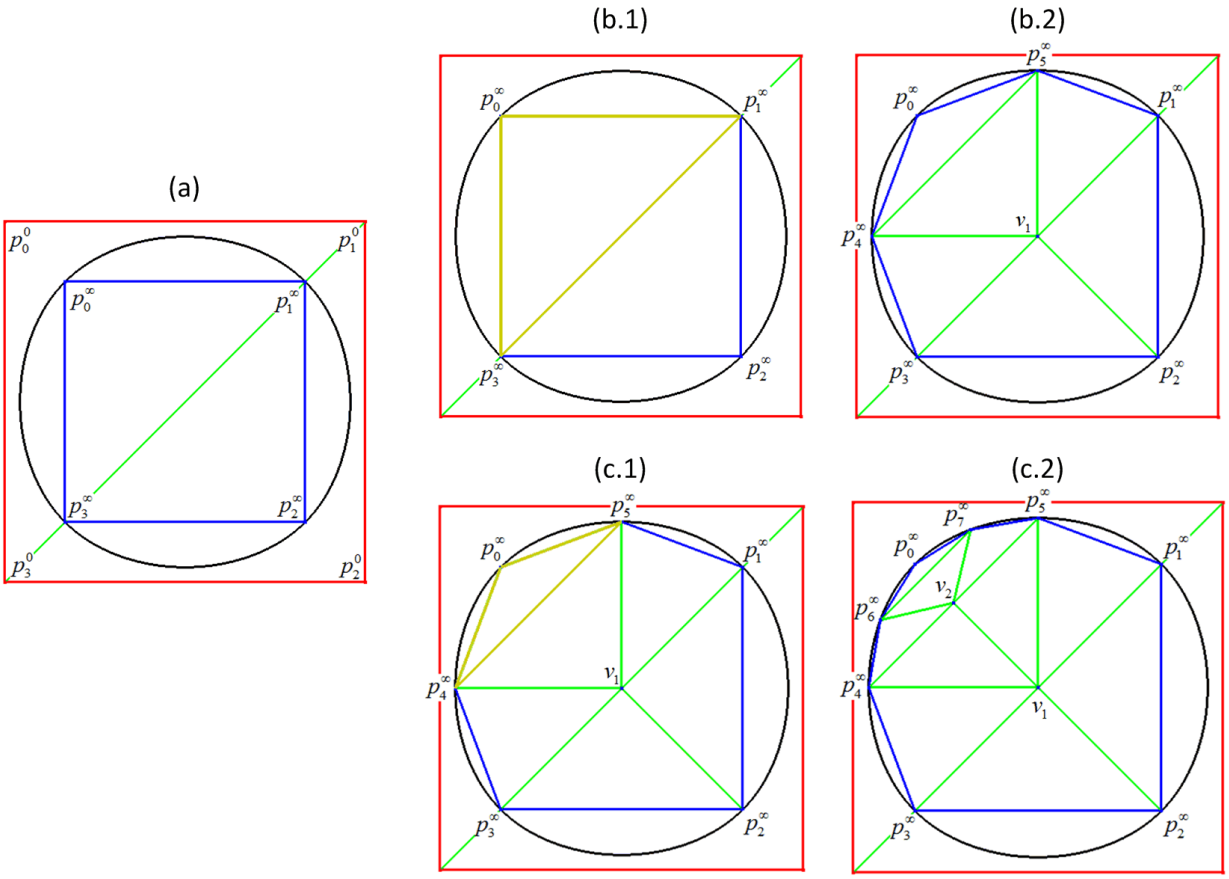

Figure 4.6: (a) The initial square curve example in red together with its limit subdivision boundary mesh in black. (b) Locally selecting the triangular face $\Delta p_0^\infty p_1^\infty p_3^\infty$ for split and the corresponding refined result. (c) Locally selecting the triangular face $\Delta p_0^\infty p_4^\infty p_5^\infty$ for split and its corresponding refined result.

The new refined mesh $M^{\sim}$ begins with four limit vertices $p_0^\infty, p_1^\infty, p_2^\infty$ and p_3^∞. Here, the adaptive refinement is replaced by manually selecting a local triangular face for split during each refinement step. First we select a triangle $\Delta p_0^\infty p_1^\infty p_3^\infty$ on the refined mesh for subdivision (see Figure 4.6(b.1)). Then the selected triangle is split into four smaller triangular faces, where the

two new vertices p_4^∞ and p_5^∞ are inserted on the corresponding boundary mesh and the one new vertex v_1 is inserted in the interior surface (see Figure 4.6(b.2)). Here we need to remark that the insertion of v_1 sets off the edge $\left(p_1^\infty, p_3^\infty\right)$ being tagged as 'split'. The neighbouring triangular face $\Delta p_1^\infty p_2^\infty p_3^\infty$ is consequently split into other two smaller triangular faces according to this tag information. Next, we locally select the triangular $\Delta p_0^\infty p_4^\infty p_5^\infty$ for refinement (see Figure 4.6(c.1)). Then this refinement results in that the selected triangle $\Delta p_0^\infty p_4^\infty p_5^\infty$ is split into four smaller triangular faces and its direct neighbouring triangular face is split into two other smaller triangular faces, where the other two limit vertices p_6^∞ and p_7^∞ are inserted on the refined boundary curve and one interior edge vertex v_2 is inserted in the interior surface (see Figure 4.6(c.2)).

Synchronously, we check up the changes related to mesh storage. Initially, there is a mesh containing 4 points and 2 triangles. Once refinements begin, a void mesh is introduced to store the refined mesh. After the first refinement, the refined mesh is updated into the mesh containing 7 points and 6 triangles. After the second refinement, the refined mesh is updated into the mesh containing 10 points and 12 triangles. The refinement changes are directly updated into the refined mesh, which does not require a hierarchy structure to store these two meshes. These results confirm the statement about the single-level refinement method in Section 4.1.5.

4.2.2 Configuration of Triangle Split Patterns

From above, we can observe that adaptive refinements on triangular surfaces are also the edge-based subdivision. During each refinement step, the triangular faces qualifying for the refinement criteria are selected for split. Then all the related edges of the qualified triangular faces are tagged as split. As the edge 'split' information is shared by the neighbouring triangular faces, the real-time check of the updated 'split' information is essential not only after each refinement step but also during the triangle splits. To avoid abrupt disconnections or cracks on adaptively refined triangular meshes, we subdivide again all the triangular faces with the edges marked as 'split'.

According to the number of the edges tagged as 'split' (see the examples (a.1), (a.2) and (a.3) in Figure 4.5), the triangle splits are categorized into two split patterns: 1-to-2 split and 1-to-4 split (see Figure 4.7(b.1) and Figure 4.7(b.2)). The number of split edges varies from 0 to 3, thus

there are four possibilities if we include the ‘no split edge’ case. In the case of two split edges in one triangle, we insert one new vertex in the middle of the opposite edge. This insertion is to avoid creating irregular triangles with undesired aspect ratios.

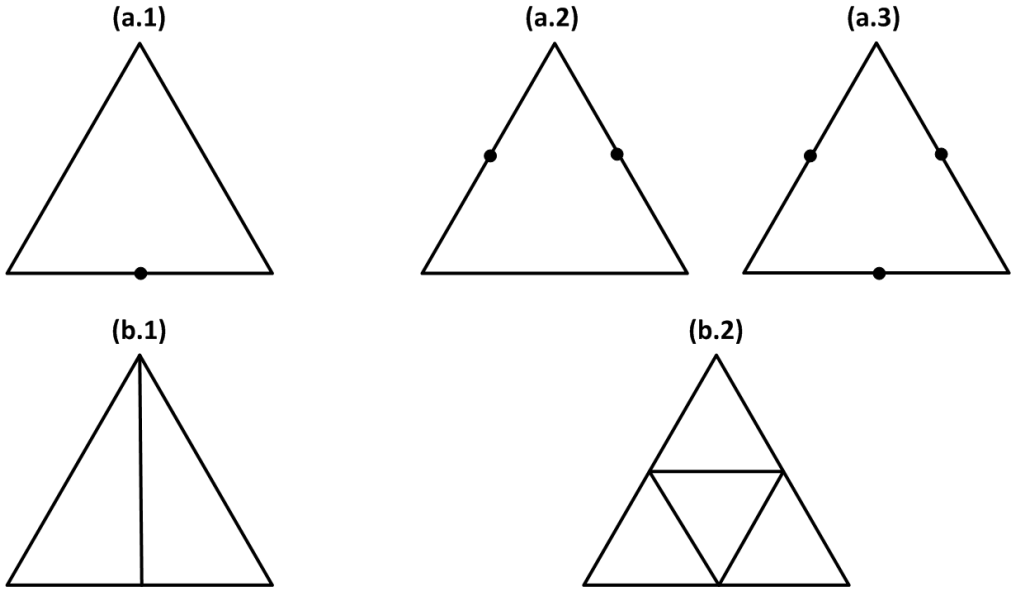

Figure 4.7: (a.1) One edge is tagged for split; (a.2) Two tagged edges; (a.3) three tagged edges are tagged. (b.1) 1-to-2 triangle split pattern; (b.2) 1-to-4 triangle split pattern.

4.2.3 Algorithms to Adaptively Refine Triangular Surfaces

The edge-based subdivision consists to insert a new vertex in the middle of the ‘tagged for split’ edge, whose algorithm is described in the Function *CreateEdgePo*int() (see Table 4-3).

Before performing this vertex insertion, we need to get two types of information: 1) is this edge on a boundary curve? 2) if yes, which segment on the initial curve does this edge correspond to?

Table 4-3: The pseudo codes for inserting a new vertex in the middle of the 'tagged for split' edge (p_1, p_2) on the refined mesh

```
Function CreateEdgePoint(Point *p1, Point* p2, bool isBoundary)
{
        // If this new vertex is on the boundary curve, it will be projected to its limit
        if(isBoundary)
        {
                int Index_OriginalEdge = p1-> LocateInitialSegment();
                //By default, this new vertex is inserted in the middle
                float ratio = 0.5f;
                float u1 = p1->u();
                float u2 = p2->u();
                //The special case for original points
                if(u2 == 0.0f)
                        u2 = 1.0f;
                float u = u1 * ratio + u2 * (1.0f - ratio);

                edgePoint = EvaluateArbitraryVertex(u, Index_OriginalEdge);
                edgePoint-> OriginalSegment () = Index_OriginalEdge;
                edgePoint->u() = u;
        }
        else
        {
                edgePoint= new Point(*p1 * 0.5f + *p2 * 0.5f);
        }
        points.push_back(edgePoint);
        return edgePoint;
}
```

In order to know if this edge is on the boundary, we need to check the triangles which contain this edge. If there is no neighbouring triangle sharing the same edge, then this edge is on the boundary. One triangle has a list of 3 neighbouring triangles. Each list element is a direct pointer to the neighbouring triangle. If this list is NULL, this triangle contains one boundary edge.

Once this edge segment is confirmed to belong to the boundary, firstly we need to locate its initial edge segment. This segment information is stored in each vertex, which can be obtained by applying the sub-function *LocateInitialEdgeSegment*() presented in Section 4.1.3. According to the location of the original edge segment, we obtain four related control vertices on the initial boundary curve mesh. Then by using the parameter value u, we split the segment. As usually,

the value u is set as 0.5 and we bisect the edge segment (p_1, p_2). Finally, by using the sub-function *EvaluateArbitraryVertex*() described in the section 4.1.3, we project the newly inserted boundary vertex to its limit.

Once this new vertex is created, its structure contains its coordinates, the issued u value and the information of its segment of origin. Here, this value u is issued once at the vertex creation, since it is essential for the further limit position calculations of other neighbouring vertices.

4.2.4 Results of Adaptive Surface Refinements

In this section, we present the results of adaptively refining the same surface mesh above. Here, the surface refinement criteria are manually defined for demonstrating the results of implementing the algorithms presented in Section 4.3.3.

In Figure 4.6, we demonstrate the results of simultaneously bisecting all the interior and boundary edges of a surface mesh. The illustrations in Figure 4.8 respectively come from performing the global bisection function in sequence on the same initial control mesh illustrated in Figure 4.6(a).

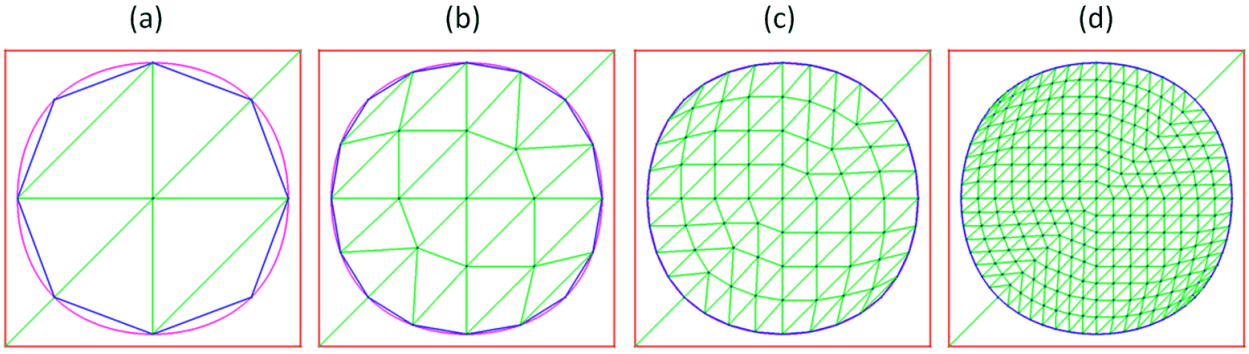

Figure 4.8: The results of bisecting all surface edges once, twice, three and four times.

In Figure 4.9, we demonstrate the results of splitting some selected triangular faces. The selected triangular faces are marked in brown, which could be interior or on the boundary. By illustrating their respective results, we can observe the changes around their neighbouring facets. On the left side of Figure 4.9, we randomly select the triangular faces. On the right side of Figure

4.9, we demonstrate their corresponding split results. The refinements are applied on the same initial control mesh illustrated in Figure 4.6(a).

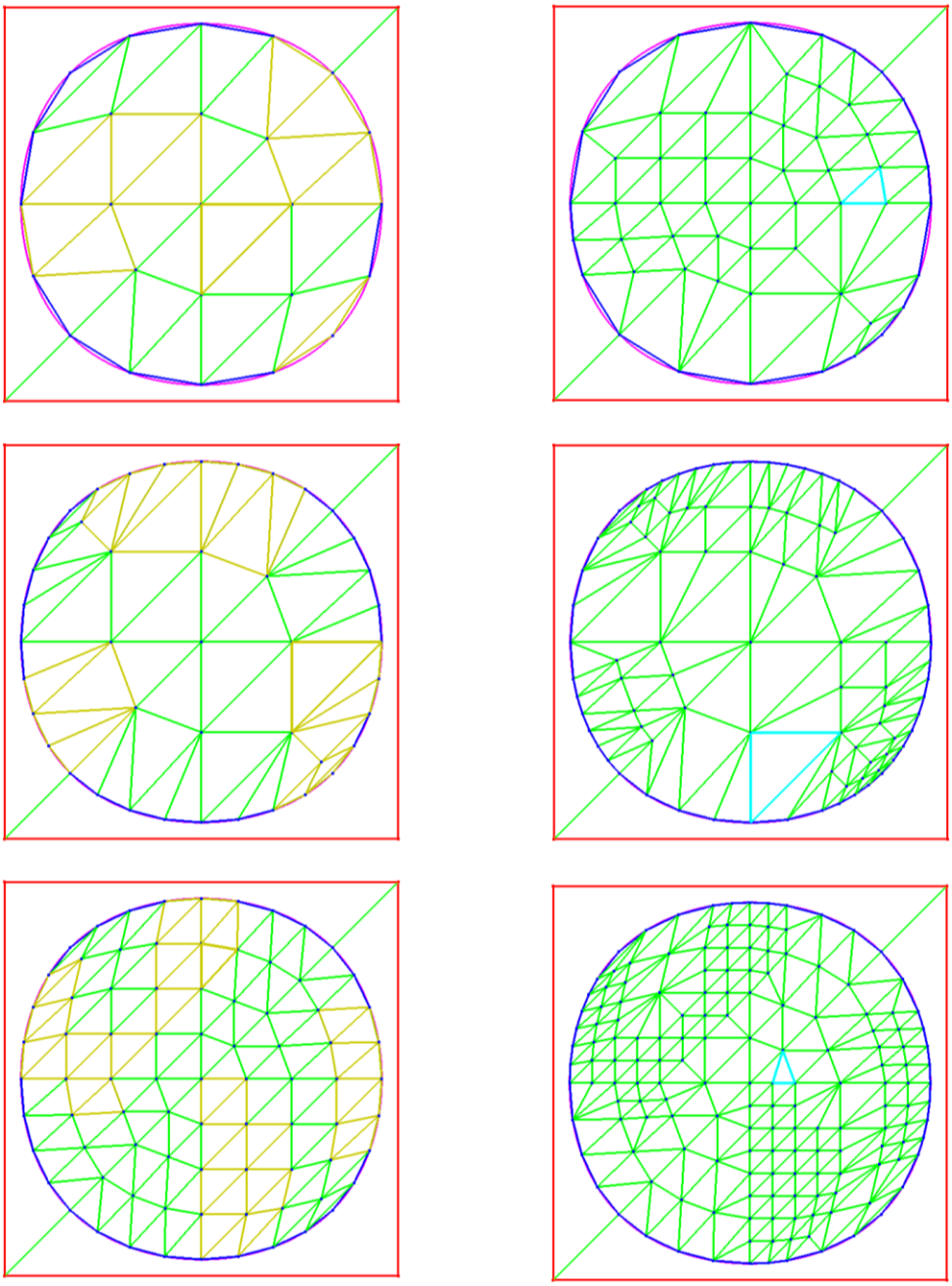

Figure 4.9: The illustrations of refining the selected triangular faces: on the left side, we mark the selected triangular faces in brown color; on the right side, we illustrate the refined results.

In the context of numerical simulations, point sources are used for analytic-based refinements. Here, we define a fixed point $(-0.9, 0.9)$ in the interior of the square curve (see Figure 4.6(a)), the coordinates of whose four initial control vertices $\{p_0^0, p_1^0, p_2^0, p_3^0\}$ are respectively $(-1.0,1.0)$, $(1.0,1.0)$, $(1.0,-1.0)$, $(-1.0,-1.0)$. The fixed point is near the control vertex p_0^0. Then we define the refinement criterion by using Equation 4.8:

$$h > c * d^2, \tag{4.8}$$

where h represents the length of the queried edge segment (p_i, p_{i+1}), c represents a constant default value and d represents the distance between the middle point of the queried edge and the fixed point.

The adaptively refined mesh results are displayed in Figure 4.10 (5 rows and 4 columns). We respectively list the mesh results by varying the value c from 0.1, 0.2, 0.5 to 0.7 in consecutive columns. Then according to the number of refinement iterations, in consecutive rows we correspondingly list the results obtained by applying the same refinement criterion. Each refined triangular surface mesh consists of two parts: the boundary mesh (illustrated in blue curves) and the interior surface (illustrated in green lines).

From these refinement results, we observe that as the refinement iteration is increased, the mesh area around the fixed point is refined more and the accuracy of the generated boundary meshes is simultaneously increased. In summary, our single-level refinement method can be successfully used for adaptively refining triangular surfaces. The boundary meshes of refined surfaces are exactly on their subdivision limits and the LODs of surface meshes can be effectively adjusted according to the refinement criteria, meanwhile the user requirements from real applications can be interpreted into refinement criteria.

In the following section, we are going to present the implementation of the single-level refinement method for adaptively refining tetrahedral meshes. The tetrahedral mesh refinement work is the core of our research. The two sections above serve as the preparation for the next section.

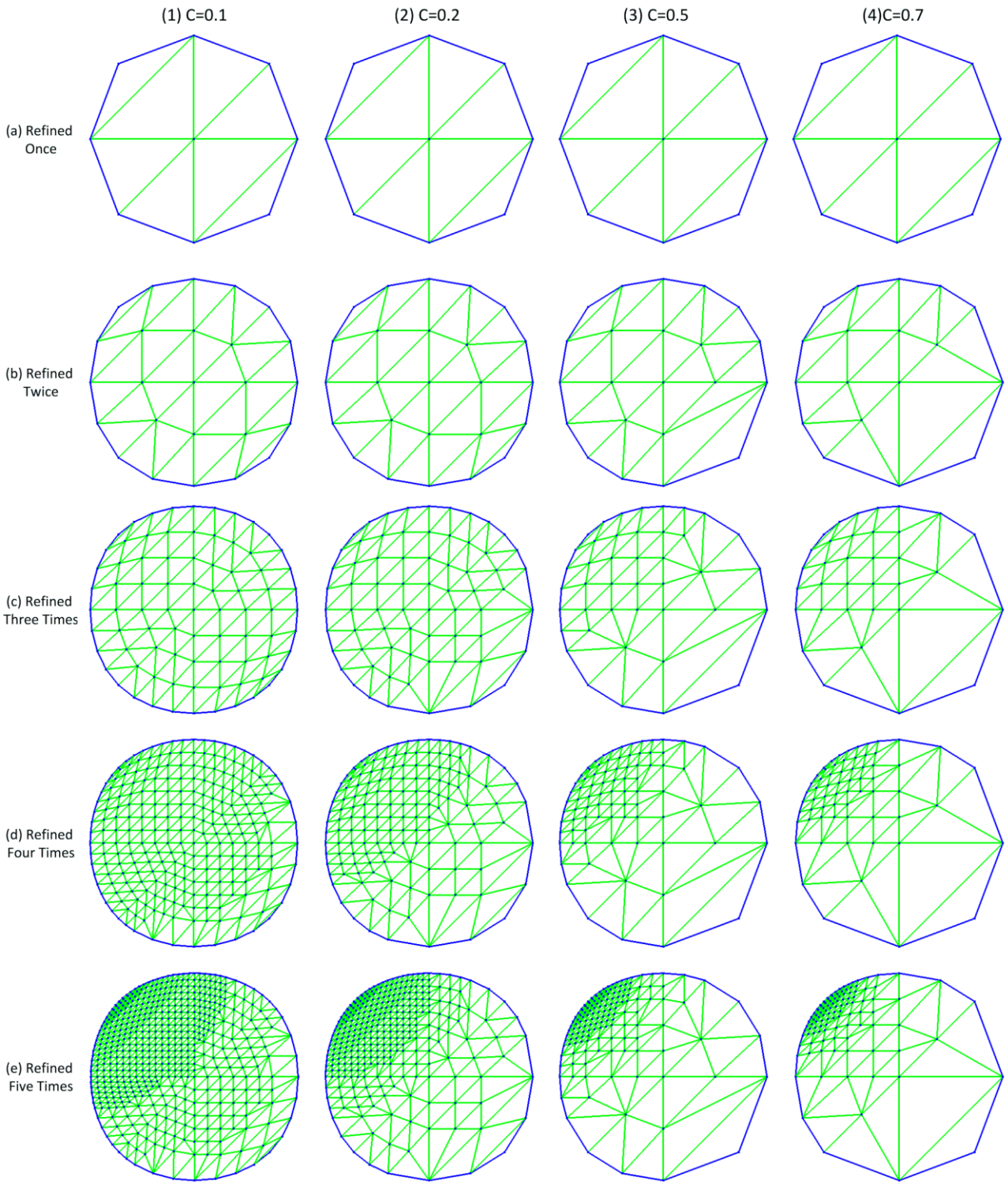

Figure 4.10: Adaptively refining a triangular mesh of a square curve by varying the constant values c.

4.3 Adaptive Refinements on 3D Tetrahedral Meshes

Due to the topological complexity and the geometrical data increase, the refinement works on tetrahedral meshes are much complicated. The advantages of naturally integrating solid subdivisions with adaptive refinements through utilizing the single-level method are much evident, when adaptive subdivisions are being performed on tetrahedral meshes, particularly those with large volume and arbitrary topology.

4.3.1 Adaptive Edge-based Tetrahedral Subdivision

Here, 3D volume meshes are still treated as two parts: the boundary and the interior. For a tetrahedral mesh, its boundary is a shell surface embedded with triangular faces, and its interior is filled with tetrahedrons (see Figure 4.11).

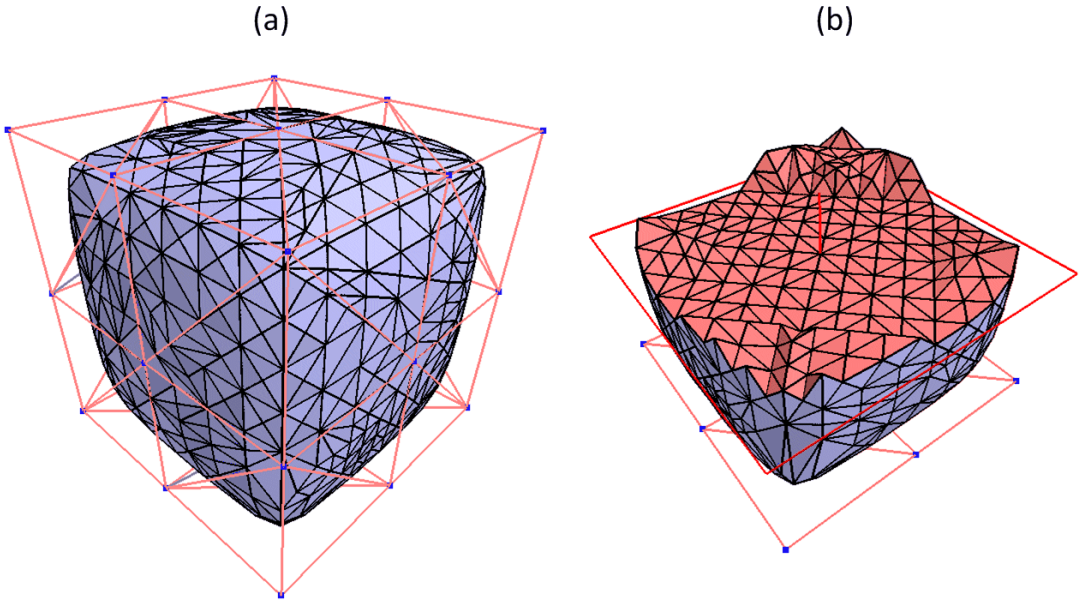

Figure 4.11: (a) The tetrahedral mesh of a cube: its initial control mesh is displayed with red lines and its adaptively refined mesh is displayed in shading mode. (b) The cube profile displayed with a cut plane: its interior tetrahedrons are in red shading and its boundary surface is in purple blue shading mode.

In our adaptive refinement work, boundary surfaces are treated as subdivision surfaces, where both the topological triangle splits and the geometrical smoothing are applied on the qualified triangular faces. In the other words, on boundary surfaces, triangle splits result in adaptively inserting vertices on the edges of boundary facets, which are directly projected onto their boundary surface limits by using the subdivision surface evaluation techniques. Meanwhile, in the tetrahedral interior, the qualified tetrahedrons are simply treated with the topological tetrahedron splits, which results in generating more split tetrahedrons.

In fact, the essence of our adaptive refinement is that 1) firstly, select all the fundamental subdivision blocks meeting the pre-defined subdivision criteria, 2) and then tag all the related edges for split, which is to insert a new vertex in the middle of the tagged edge, 3) finally, if the newly inserted vertices are on boundaries, the evaluation masks will be applied to insure that they are on their limit positions. In practice, if the subdivision criterion is based on the edge evaluation, such as curvature or edge length, all the qualified edges will be directly selected during the first step. On the whole, the refinements on tetrahedral mesh are the edge-based subdivision, and their fundamental subdivision blocks are the tetrahedrons (T_i).

4.3.2 Mesh Connectivity

As what we have presented in the section 3.1, for a tetrahedral mesh, the key to differentiate the boundary and the interior is mesh connectivity. As the three basic geometric elements on tetrahedral meshes: *Vertex* (or *Control Point*) (V_i), *Face* (F_i) and *Tetrahedron* (T_i), their corresponding relations determine the connectivity. In our work, the input data of the initial control mesh consist of the information based on the three elements: the coordinates of vertices, the list of faces and the list of tetrahedrons.

In order to clarify the relations between these three different elements, we demonstrate a tetrahedral mesh example, which contains 10 vertices, 24 faces and 8 tetrahedrons (see Figure 4.12). Its input data information including the list of vertices' coordinates, the list of faces and the list of tetrahedrons, is presented in Figure 4.13.

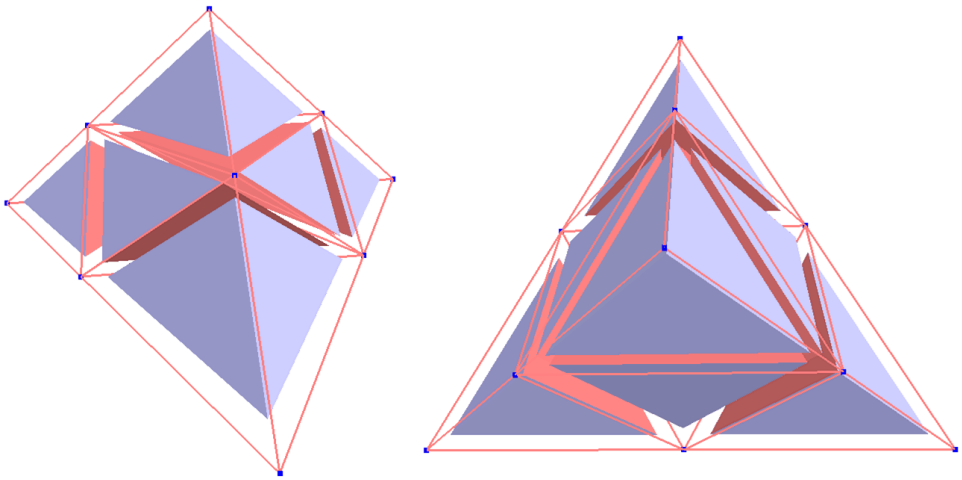

Figure 4.12: The tetrahedral mesh example in two different viewpoints.

(a)

Vertex No.	Coordinates
0	(0.0, 0.0, 1.0)
1	(1.0, 0.0, 0.0)
2	(0.0, 0.0, 0.0)
3	(0.0, 1.0, 0.0)
4	(0.5, 0.0, 0.5)
5	(0.5, 0.0, 0.0)
6	(0.0, 0.0, 0.5)
7	(0.0, 0.5, 0.5)
8	(0.5, 0.5, 0.0)
9	(0.0, 0.5, 0.0)

(b)

Face No.	$[V_1\ V_2\ V_3]$	Face No.	$[V_1\ V_2\ V_3]$
0	[0 4 7]	12	[4 8 5]
1	[4 6 7]	13	[8 1 5]
2	[6 0 7]	14	[1 4 5]
3	[0 6 4]	15	[4 1 8]
4	[7 3 9]	16	[6 8 7]
5	[3 8 9]	17	[8 4 7]
6	[8 7 9]	18	[6 4 8]
7	[7 8 3]	19	[5 8 9]
8	[6 9 2]	20	[8 6 9]
9	[9 5 2]	21	[5 6 8]
10	[5 6 2]	22	[6 7 9]
11	[6 5 9]	23	[4 5 6]

(c)

Tetrahedron No.	$< F_1\ F_2\ F_3\ F_4 >$
0	< 0 4 6 7>
1	< 7 3 9 8>
2	< 6 9 5 2>
3	< 4 8 1 5>
4	< 6 8 4 7>
5	< 5 8 6 9>
6	< 8 7 6 9>
7	< 4 6 5 8>

Figure 4.13: The input data information of one tetrahedral mesh example: (a) the list of vertices' coordinates; (b) the list of faces; (c) the list of tetrahedrons.

In general, the computation of mesh connectivity includes: 1) setting up the neighbouring information; 2) tagging the boundary information. For each Tetrahedron T_i, we add them as neighbor of each of their 4 composing Faces. The boundary information of a tetrahedron is

obtained through its children elements: four neighbouring faces. The pseudo code for adding the Face neighbor information of all Tetrahedron T_i is presented in Table 4-4:

Table 4-4: The pseudo code for adding the Face neighbor information of all Tetrahedron T_i

```
For (Each Tetrahedron T_i)
{
        For (Each Face F_i of T_i)
        {
                Add T_i as tetrahedron neighbor of F_i;
        }
}
```

The element *Face* represents one side of its parent element: *tetrahedron*. Basically, each *face* (F_i) is constructed with three vertices $\left[V_i, V_j, V_k\right]$. If one *face* is not shared by two tetrahedrons, this face will be flagged as being on the boundary surfaces. Since the boundary triangular faces are essential for calculating the limit subdivision surface, the related variables such as: 3 sets of parameterization values (u, v) used for locating the position of any newly inserted edge vertex on a boundary triangular face and a pointer to the triangular face on initial boundary surface meshes are pre-stored in the element *face*. The pseudo code for tagging one triangular face as a boundary surfaces is listed in Table 4-5.

Table 4-5: The pseudo code for tagging one triangular face as a boundary surfaces

```
For (Each Face Fi)
{
    If (Fi has only one neighbouring tetrahedron)
    Then
    {
        Fi.(The boundary flag) = true;
        <[Vi,Vj,Vk] of Fi>.(The boundary flag) = true;
        <The neighbour Ti of Fi >.(The boundary flag) = true;
        For (Each Vertex in [Vi,Vj,Vk] of Fi)
        {
            Add the other 2 Vertices as the neighbouring boundary vertices to the
            current Vertex;
            Add the Face Fi as the neighbouring boundary face to the current
            Vertex;
        }
    }
    Else
    {
        Fi.(The boundary flag) = false;
        Add each neighbouring tetrahedron of Fi as neighbor of each other
    }
    For (Each Vertex in [Vi,Vj,Vk] of Fi)
    {
        Add the other 2 Vertices as the neighbouring vertices to the current Vertex;
    }
}
```

Evidently, in the interior of tetrahedral meshes, each tetrahedron is an individual geometric element. And the interior mesh connectivity is defined by the relation between any two tetrahedrons (T_i, T_j). Likewise, on boundary surfaces of tetrahedral meshes, each triangular face is an individual geometric element. Thus, the mesh connectivity on boundary surfaces is defined by computing the relation between any two triangular faces (F_i, F_j) (see Table 4-6).

Table 4-6: The pseudo code for computing connectivity between F_i and F_j

```
For (Each Face F_i)
{
    If (F_i.(The boundary flag) = true)
    Then
    {
        For (Each Vertex V_i of F_i)
        {
            For (Each Face F_j of V_i 's neighbor boundary faces)
            {
                If (F_i is neighbour of F_j)
                {
                    Add F_i as the boundary neighbour of F_j;
                    Add F_j as the boundary neighbour of F_i;
                }
            }
        }
    }
}
```

4.3.3 Three Tetrahedron Split Patterns

Once mesh connectivity is set up, the smoothing operations will be applied only on boundary surfaces, while the topological splits are applied on all the qualified tetrahedrons. Normally, a tetrahedron split is to insert a new vertex in the middle of each tetrahedral edge, which generates 8 smaller tetrahedrons. This tetrahedral split is the 1-to-8 tetrahedron split presented in Section 3.1.1.

Here, what needs to be mentioned is that if the subdivision criterion is based on tetrahedron volume, it will be the 1-to-8 splits that are primitively performed on the tetrahedrons meeting the subdivision criteria. Since the subdivision algorithms in our refinement works are based on edges whose split information are shared by the other neighbouring tetrahedrons, we introduce another two tetrahedron split patterns: 1-to-4 split (see Figure 4.14(a)) and 1-to-2 split (see Figure 4.14(b)) to split all the neighbouring tetrahedrons with the edges marked as ‘split’. This is to avoid abrupt disconnections or cracks on adaptively refined tetrahedral meshes. The 1-to-4

tetrahedron split pattern is to split the current tetrahedron into 4 smaller tetrahedrons. The 1-to-2 tetrahedron split pattern is to split the current tetrahedron into 2 smaller tetrahedrons and the split direction is decided by the edge tagged for split. The algorithm of tetrahedron splits in the three patterns is described in Table 4-7.

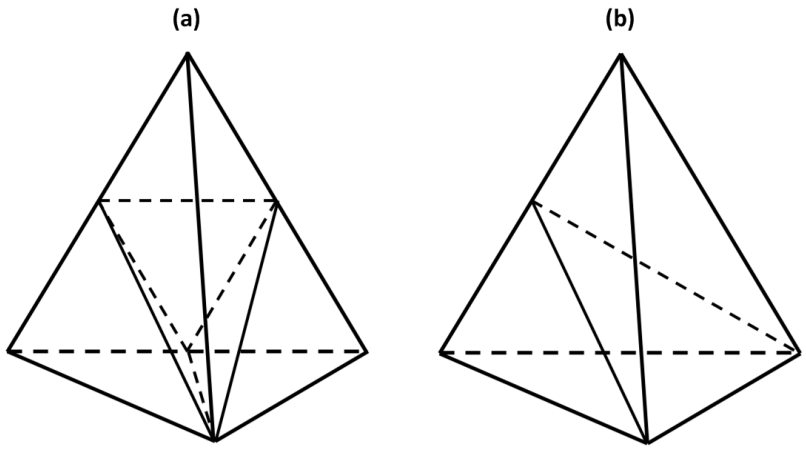

Figure 4.14: The configuration of tetrahedron split patterns: (a) 1-to-4 split; (b) 1-to-2 split.

Each tetrahedron has 6 edges, and then usually, for all tetrahedrons on one tetrahedral mesh, the edges tagged for 'split' vary from 0 to 6. However, if 4 or 5 edges of one tetrahedron are tagged as 'split', it will generate many irregular tetrahedrons. In this case, our solution is to apply a full tetrahedron split (1-to-8 tetrahedron split) on the corresponding tetrahedron. In other words, we consider this case as the one with all 6 edges tagged for 'split' through marking the other 2 edges or 1 edge as 'split', which facilitates generating 8 regular tetrahedrons. Then for 3 tagged edges on the same triangular face, we apply 1-to-4 tetrahedron split. Finally, all the remaining cases are dealt with cyclic 1-to-2 tetrahedron splits. Since the tagged information of the neighbouring tetrahedrons is being interactively changed during adaptive subdivision, this information of all the tetrahedrons including the new ones generated in real time must be updated all the time. That's why from Step 1 to Step 5 in Table 4-7 are five independent loops, which permits promptly updating the topology of the corresponding tetrahedral mesh.

The difference among the three tetrahedron splits is that 1-to-8 split and 1-to-4 split are to split the current tetrahedrons with 6 edges tagged as 'split' or 3 edges tagged as 'split' at the same triangular face, whereas 1-to-2 split is to split all the neighbouring tetrahedrons sharing the same

edge tagged as ‘split’, where the function *SplitThisEdge*() is performed (see Table 4-8). This function is used to split each corresponding tetrahedron into two half tetrahedrons (see Figure 4.14(b)). This 1-to-2 tetrahedron splits the two neighbouring faces sharing the edge tagged as ‘split’.

Table 4-7: The algorithm of tetrahedron splits in the three patterns

```
Function TetrahedronSplitsInThreePatterns()
{
    1. To check if all tetrahedrons have 0, 1, 2, 3 or 6 edges tagged for split;
       If one tetrahedron has 4 or 5 edges tagged, then to tag all the remaining edges as
       ‘split’.
    2. To loop on all the tetrahedrons and then according to the number of edges tagged as
       ‘split’, categorize them for the later splits:
       switch(TheNumberOfEdgesTaggedAsSplit)
       {
              case 0:
                     When 0 edge is tagged as ‘split’, it means nothing to do;
              case 1 or 2:
                     When 1 or 2 edges are tagged as ‘split’, it will be handled by the
                     function SplitThisEdge();
              case 3:
                     When 3 edges are tagged as ‘split’, it is a special case. In this case,
                     firstly we need to check if the 3 tagged edges are on the same
                     triangular face. If so, we can apply a 1-to-4 split. If not, it will be
                     handled by the function SplitThisEdge();
              case 4 or 5:
                     This case can not happen, because during the 1st step, any
                     tetrahedron with 4 or 5 edge tagged as ‘split’ has been updated as the
                     one with 6 edges tagged as ‘split’;
              case 6:
                     1-to-8 full tetrahedron split, which generates 8 new tetrahedrons;
       }
    3. Then we loop all the tetrahedrons and subdivide those marked as 1-to-8 split.
    4. Then we loop all the tetrahedrons and subdivide those marked as 1-to-4 split.
    5. Then we loop all the tetrahedrons and subdivide those marked as 1-to-2 split.
    6. Finally, we recompute the connectivity.
}
```

From Table 4-8, we can find that there is a new function *CreateEdgeVertex*() (see Table 4-9) involved in the function above *SplitThisEdge*(). Since this new function is equally essential for

the subdivision operators on boundary surfaces of tetrahedral meshes, we first present the related splits on triangular faces on boundary surfaces.

Table 4-8: The pseudo codes for splitting the current edge with two endpoints (A, B)

```
Function SplitThisEdge(VertexPtr A, VertexPtr B)
{
    Insert a new Edge Vertex C in the middle of A and B through calling the function
    CreateEdgeVertex();
    For (Each T_i in A)
    {
        If (T_i NOT contains B)
        {
            Skip to the next T_i;
        }
        Get the two untouched faces of T_i (FU1, FU2);
        Get the two half split faces of T_i (FH1, FH2);
        Create the new face FM in the middle of (FU1, FU2);
        If (FH1.(The boundary flag) = true
        {
            Get the (u, v) coordinates of the middle edge of FH1;
        }
        Create the two half faces (FH1_1, FH1_2) from FH1;
        If (FH2.(The boundary flag) = true)
        {
            Get the (u, v) coordinates of the middle edge of FH2;
        }
        Create the two half faces (FH2_1, FH2_2) from FH2;
        Create a new Tetrahedron from FM, FU1, FH1_1 and FH2_1;
        Create a new Tetrahedron from FM, FU2, FH1_2 and FH2_2;
        Keep track of T_i for later deleting;
    }
}
```

As tetrahedral mesh connectivity accounts not only for tetrahedral interior but also boundary surfaces, triangular face splits are applied during the topological splits on the tetrahedrons, which contain the boundary faces. Actually, there are two triangle split patterns: full split (see Figure 4.7(b.1)) and half split (see Figure 4.7(b.2)), as what we have presented in the section 4.2.2.

On the whole, the edge-based subdivision on tetrahedral meshes can be summarized as two parts: 1) in the interior, all the corresponding *tetrahedrons* are split into 8, 4, or 2 smaller tetrahedrons; 2) on the boundary surfaces, the triangular faces are split into 4 or 2 smaller triangular faces. At the same time, the geometric smoothing is applied to the new edge vertices on the split boundary faces. In fact, this smoothing is to project any qualified boundary surface region from its initial control mesh to its subdivision limit by using the evaluation technique on Loop subdivision surface, whose details are going to be elaborated more in the next chapter. These two combined parts permit the whole tetrahedral meshes to be adaptively refined, while their corresponding boundary surfaces to be smoothed and be directly kept on their subdivision limits.

Table 4-9: The pseudo codes for creating the new edge vertex between the two endpoints (V1, V2)

```
VertexPtr CreateEdgeVertex(VertexPtr V1, VertexPtr V2)
{
        V.position = 0.5*(V1.position + V2.position);
        For (Each Neighbouring Triangular Boundary Face NTBF_V1[i] of V1)
        {
                For (Each Neighbouring Triangular Boundary Face NTBF_V2[j] of V2)
                {
                        If (NTBF_V1[i] == NTBF_V2[j]) //The split edge is on the boundary
                        surface
                        {
                        Set V.position to its limit position using the Evaluation algorithm
                        (reference Chapter 5 and Section 2.1.4);
                        }
                }
        }
        Add V to the list of vertices;
        return V;
}
```

4.3.4 Results of Adaptive Tetrahedral Mesh Refinements

In this section, we demonstrate the results of adaptively refining tetrahedral meshes. Here, the cube tetrahedral mesh is chosen for illustrating the smoothing effect on its boundary surface. And

the Schoenhardt tetrahedral mesh is chosen for illustrating the subdivision effect on the example with arbitrary topology. The initial control meshes (in red lines), subdivision limits (in grey), and the start meshes for refinement (in purple blue) are respectively displayed in Figure 4.15.

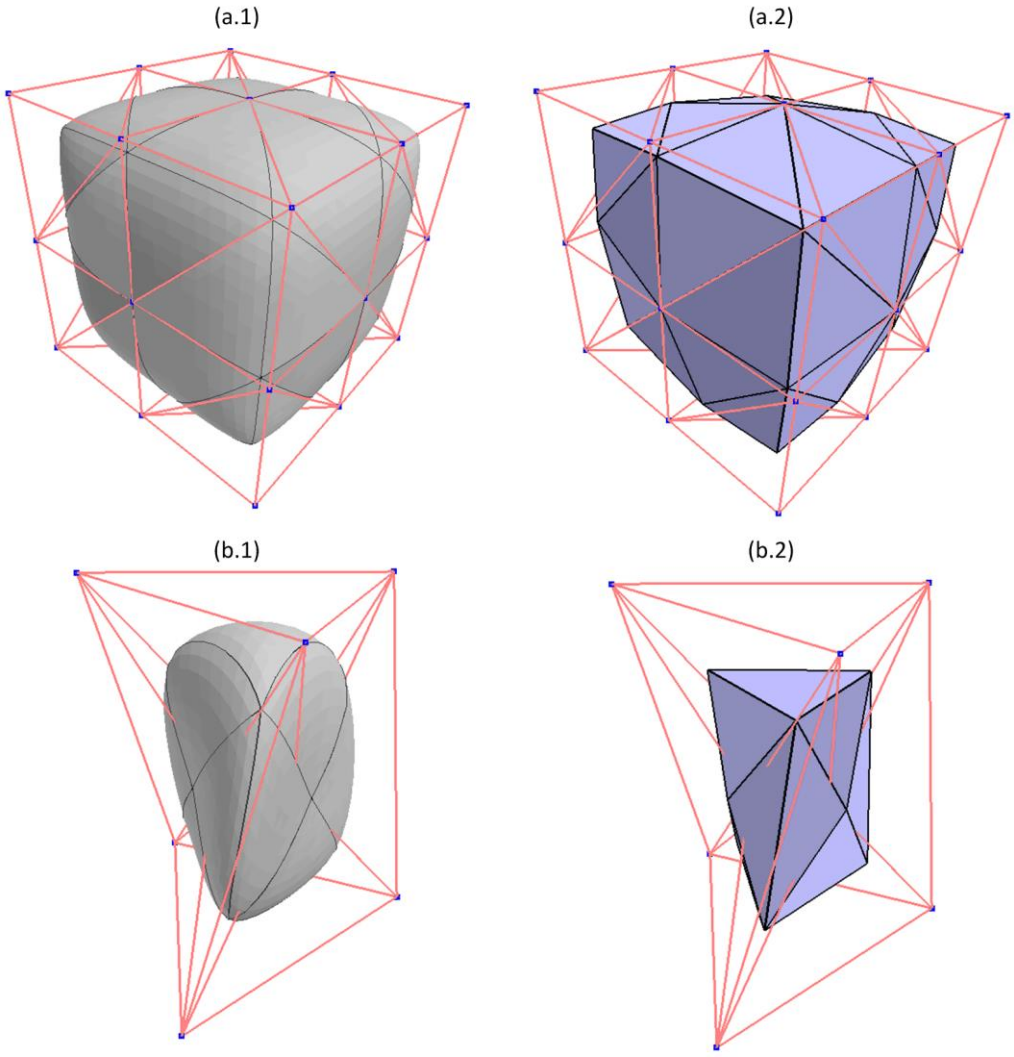

Figure 4.15: (a) a cube tetrahedral mesh example; (b) a Schoenhardt tetrahedral mesh example.

Our refinement begins with a start mesh consisting of the limit positions of initial control vertices. The first refinement criterion is based on tetrahedron volumes. If the queried tetrahedron volume is bigger than the average, we split it. Respectively, we apply the criterion based on

volume onto the two tetrahedral mesh examples above once, twice, three times, four times and five times.

The cube tetrahedral mesh refined once is displayed in blue purple shading mode in Figures 4.16(a.1), and the cut profile of this mesh is displayed in 4.16(b.1), where the cut plane is the dark red square curve around the mesh profile and the volume interior is in red shading mode. The mesh refined twice is displayed in Figure 4.16(a.2) and Figure 4.16(b.2)… etc.

In the same way, we display the refined Schoenhardt tetrahedral meshes in Figure 4.17. From the results of these two examples, we conclude that 1) solid subdivision based on subdivision surface parameterization can be successfully used for smoothing boundary surfaces of tetrahedral meshes; 2) this new solid subdivision-based refinement can be used for refining tetrahedral meshes with arbitrary topology; 3) the single-level refinement method can be equally used for adaptively refining tetrahedral meshes.

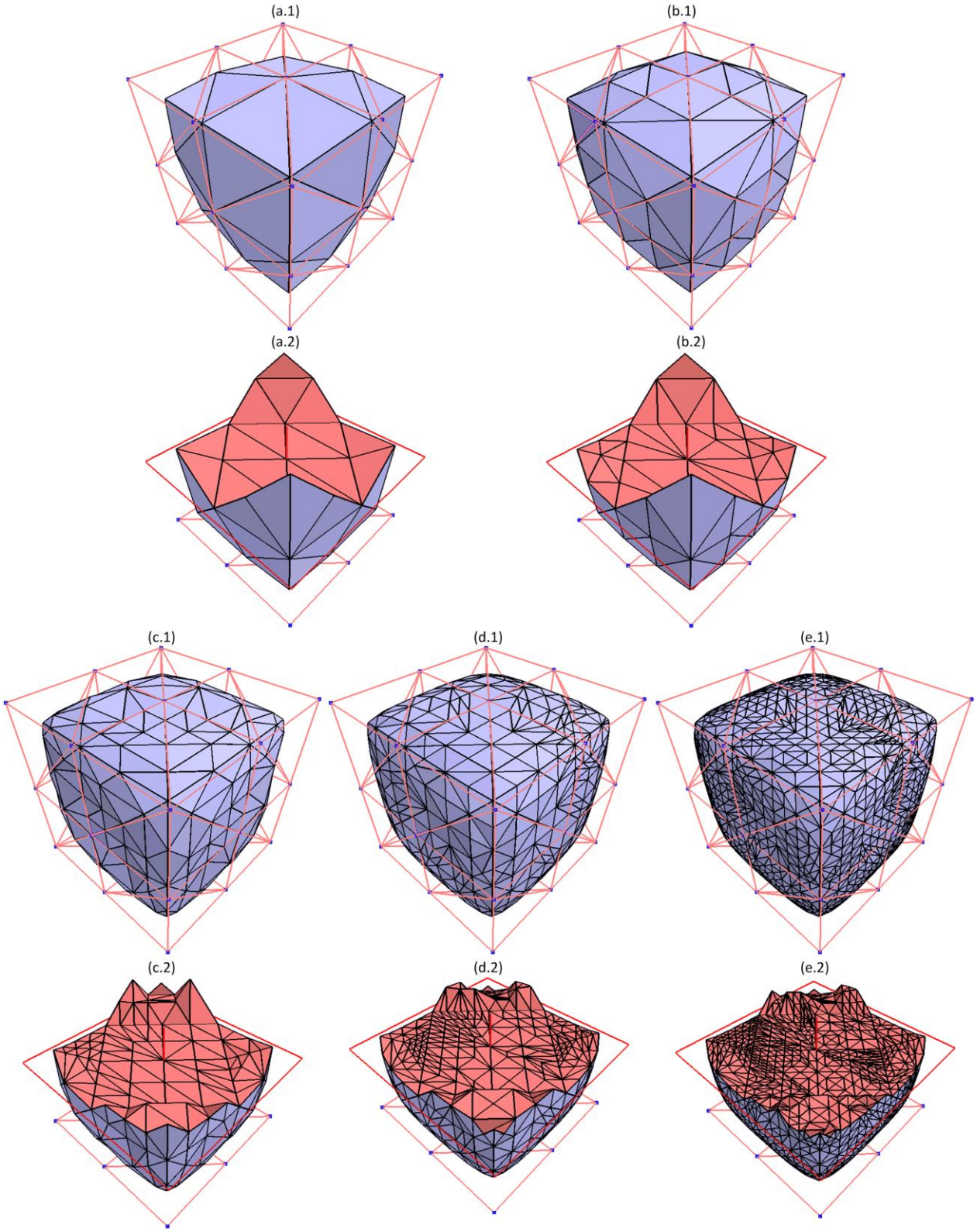

Figure 4.16: Applying adaptive subdivisions on a cube tetrahedral mesh once, twice, three times, four times and five times.

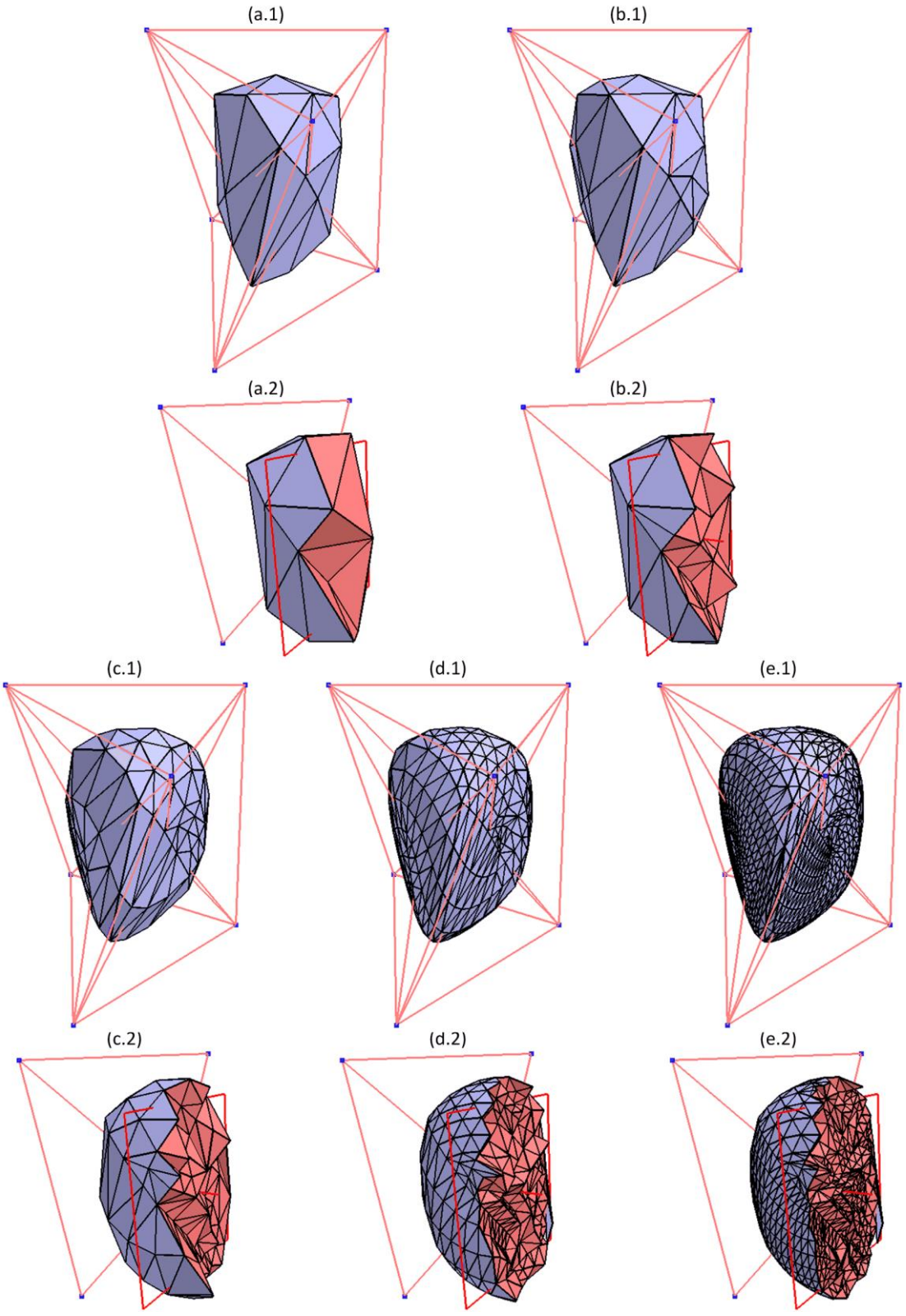

Figure 4.17: The refined cube tetrahedral meshes are obtained by applying adaptive subdivisions once, twice, three times, four times and five times.

To validate that the single-level refinement method can be equally implemented for engineering analysis, we interpret the analytic-based criterion defined in Equation 4.8 into our solid subdivision work. We apply this criterion on the same cube tetrahedral mesh example (see Figure 4.15(a)). Here, the point source is set as the vertex with the coordinate (1.0, 1.0, 1.0). This vertex is an initial control vertex on the cube tetrahedral mesh. We display the adaptively refined cube mesh results in Figure 4.18 (5 rows and 3 columns). We respectively list the mesh results by varying the value c from 0.7, 0.5 to 0.3 in consecutive columns. Then according to the number of refinement iterations, in consecutive rows we correspondingly list the refined mesh results obtained by applying the same refinement criterion.

In order to visualization the changes inside the volume interior, in Figure 4.19 we display the cut profiles of the refined tetrahedral mesh results by using cut planes. In this demonstration, we only present the refined results, which are obtained by applying adaptive subdivisions four times and five times.

The refinement results in Figure 4.18 and Figure 4.19, demonstrate that the single-level refinement method can be successfully integrated with the analytic-based criteria for engineering analysis. And the LODs of tetrahedral meshes can be flexibly controlled by adjusting the refinement criteria.

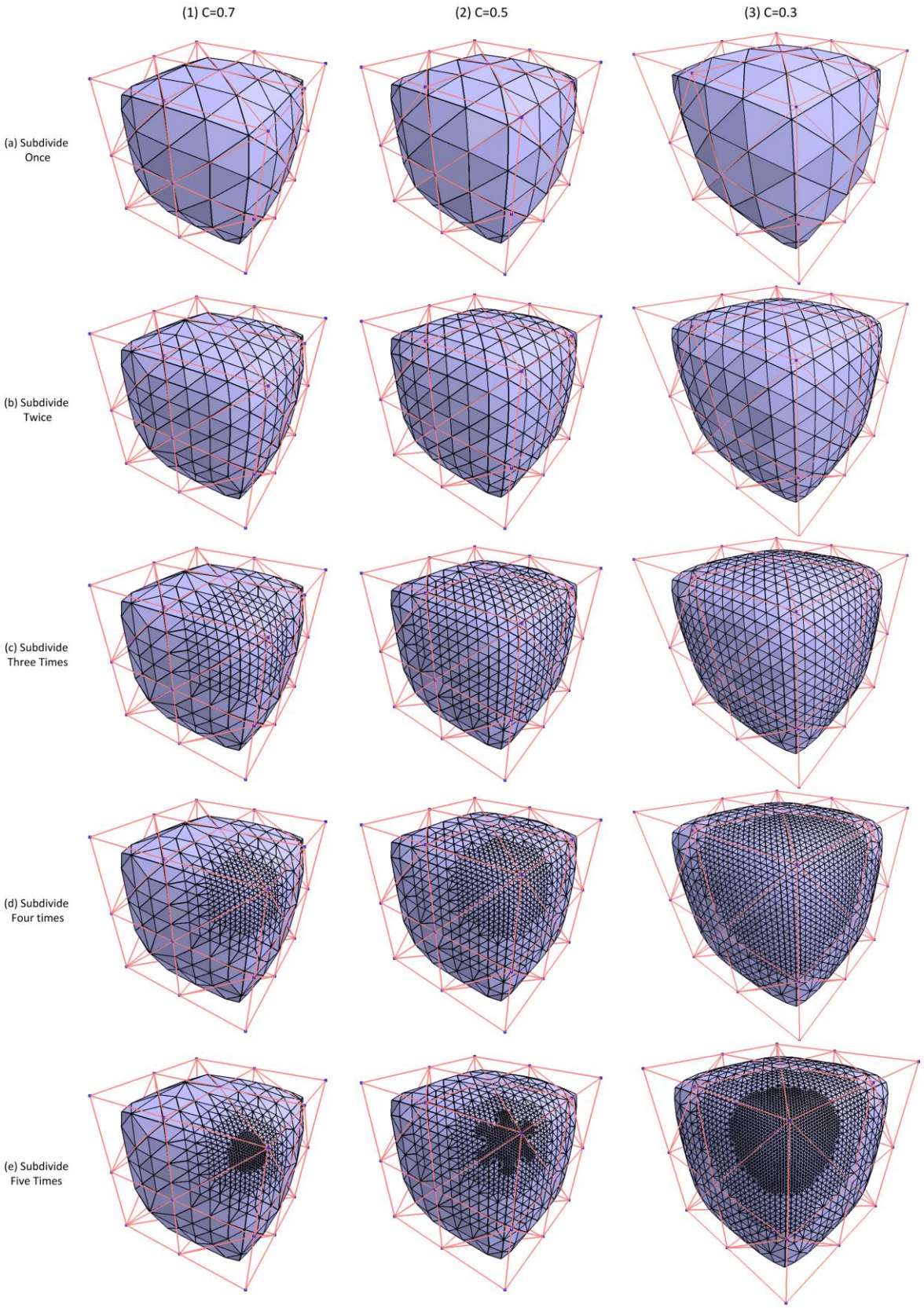

Figure 4.18: Adaptively refining the cube tetrahedral mesh by varying the constant value c.

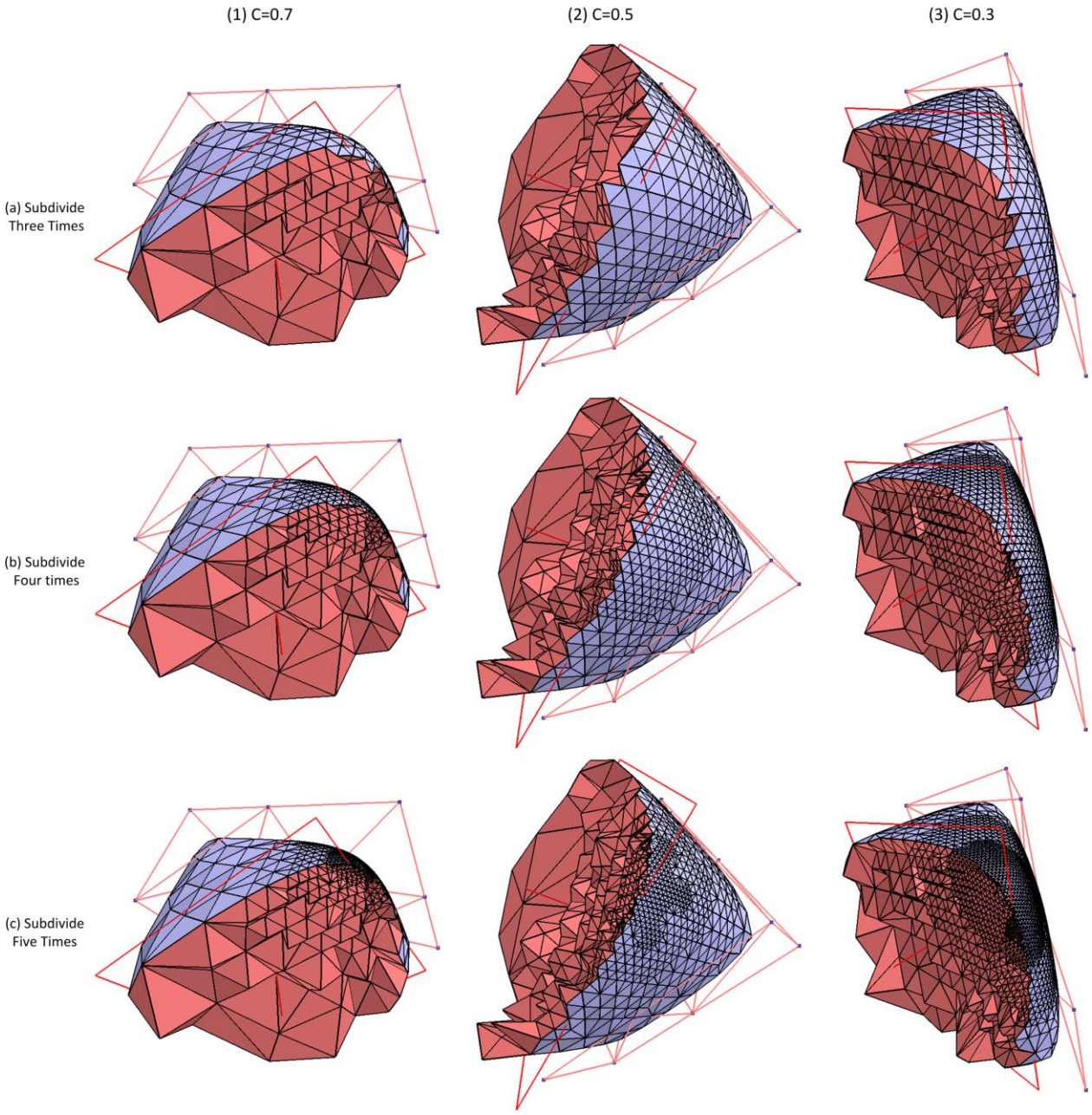

Figure 4.19: The cut profiles of the adaptively refined cube tetrahedral mesh obtained by varying the constant value c.

Regarding the mesh degeneracy issue, we compute the tetrahedron shape measure: η evaluations presented in Section 3.2.3 on the adaptively refined tetrahedral meshes of the cube

example (see Figure 4.15(a) and Figure 4.16). Here, we apply the same volume refinement criterion on the cube example once, twice, three times, four times and five times. In order to visualize the η variation on the five adaptively subdivided tetrahedral meshes, we introduce a "η display" feature, which permits displaying the tetrahedrons with varying colors. More specifically, tetrahedrons with η values close to 1 are displayed in a light white shade, and tetrahedrons with η values close to 0 are displayed in a deep black shade. Thus, the color of the displayed tetrahedron is determined by its η value. In Figure 4.20, we illustrate the refined cube mesh results together with its start mesh and display them in the "η display" mode.

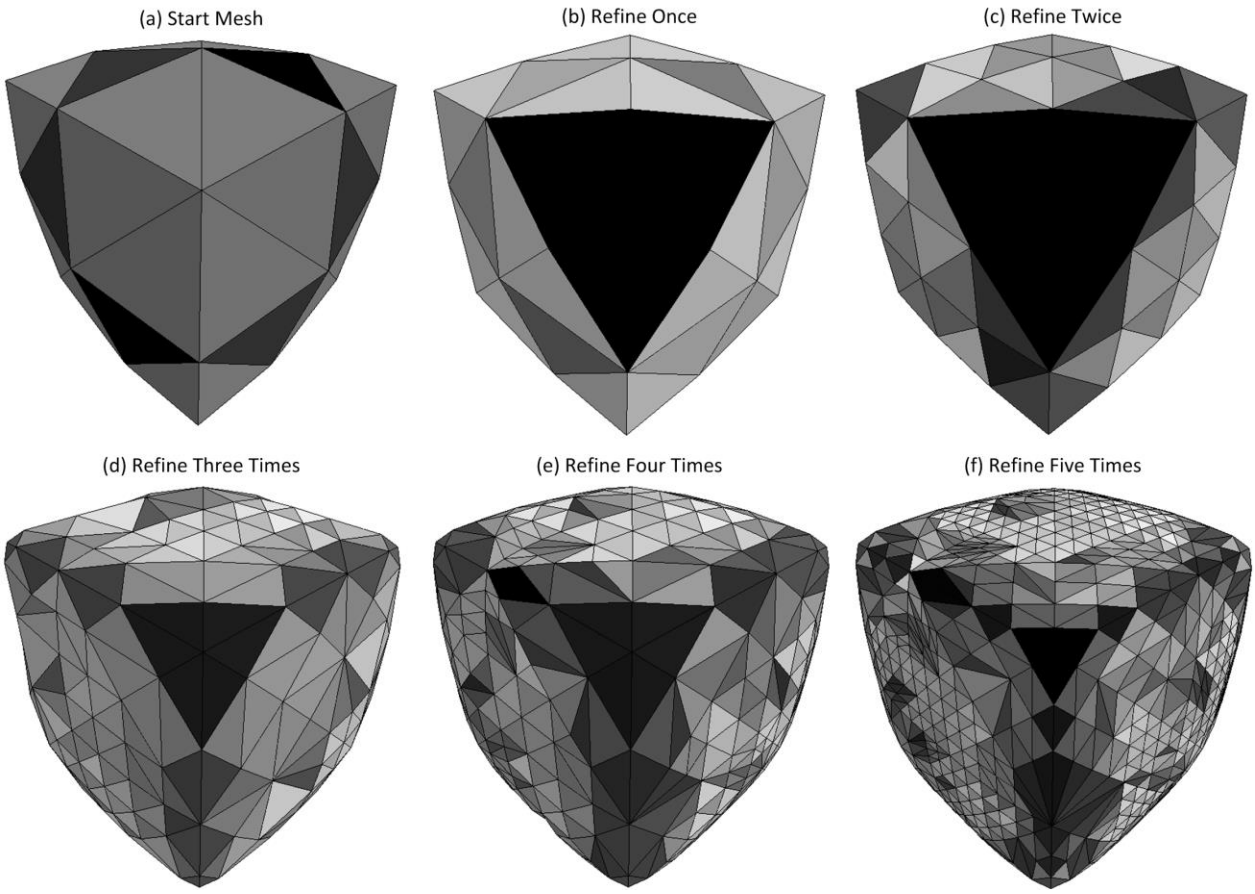

Figure 4.20: The adaptively refined cube tetrahedral meshes displayed with the "η display" mode.

In Table 4-10, we list the corresponding η evaluations on the tetrahedral mesh results obtained by adaptively applying the same volume refinement criterion.

Table 4-10: The corresponding η evaluations on the five adaptively refined cube tetrahedral meshes together with its start subdivision mesh.

Refinement / Evaluation / Times	Start mesh	Refine Once	Refine Twice	Refine Three Times	Refine Four Times	Refine Five Times
minimum η	0.687	0.391	0.391	0.089	0.097	0.084
maximum η	1.000	1.000	1.000	1.000	1.000	1.000
Average η	0.810	0.801	0.713	0.701	0.715	0.661

Regarding the η evaluation results listed in Table 4-10, we use line charts to display different η trends over the number of refinement times (see Figure 4.21).

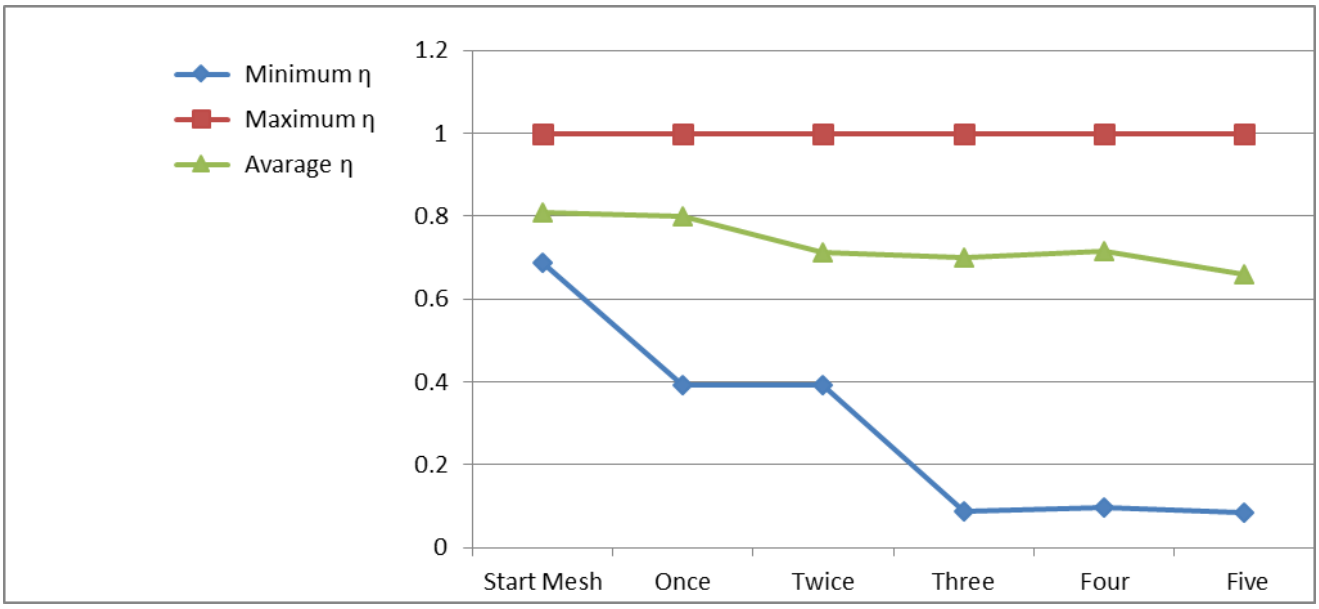

Figure 4.21: η evaluations on the five adaptively subdivided cube tetrahedron meshes together with its start subdivision mesh.

According to the evaluation results on the adaptively refined cube tetrahedral meshes, we can observe that since the beginning of adaptive refinements, the maximum η is stabilized at 1.0. The average η values gradually decrease as the refinements continue, and their range is between 0.8 and 0.6. The minimum η values decrease more rapidly than the average η values as we apply more refinements on the studied cube tetrahedral meshes. By comparison with the evaluation results on the uniformly subdivided cube tetrahedral meshes presented in Table 3-9 and Figure 3.20, we conclude that the η values obtained by evaluating the adaptively refined cube tetrahedral meshes have been improved.

Regarding the observations above, we summarize that subdivision-based refinements represent attractive aspects in smoothing boundary surfaces of tetrahedral meshes. However, according to the illustrations in Figure 4.20, the tetrahedrons on boundary surfaces could become even flatter as boundary surfaces converge toward their subdivision limits. The η variation is not constant as the number of refinement iteration changes. Here, we emphasize that adaptive subdivision can improve mesh quality, but it is not the best option for controlling mesh degeneracy. Moreover, the η values are also determined by other factors, such as the topology of initial control meshes and the refinement criteria. In conclusion, the mesh degeneracy issue is beyond our current research scope, but it could be an interesting research subject for further studying.

4.4 Summary of Adaptive Refinements

In summary, our adaptive refinement works are edge-based subdivisions. The essence of adaptive refinements is to combine topological splits and geometrical smoothing, which are detailed as edge-based splits and boundary projection in practice. The single-level method is developed to naturally merge these two operations, which eventually facilitates the integration of adaptive subdivision.

The edge-based splits are guided by concrete refinement criteria. Once an edge of a subdivision entity is marked as ‘split’, it is important to check if neighbouring entities need to be split. The neighbouring information is defined by computing mesh connectivity. The complexity

of mesh connectivity is decided by the mesh topology. Evidently, this complexity increases as the refined mesh is turned from 1D curve to 3D tetrahedral mesh.

By projecting the qualified boundary mesh parts to their subdivision limits, we obtain a refined mesh, whose boundary mesh is always on its subdivision limit. The evaluation mask presented in Equation 4.5 is utilized for performing the boundary projections on curve and triangular surface refinements. To project boundary surfaces onto their subdivision limits, we recur to the Loop subdivision limit evaluation techniques. However, as what we have presented in Section 2.1.4, Jos Stam's evaluation method has its limitation, which could not resolve the evaluations on all irregular meshes. To solve this question, we are going to discuss two other related evaluation methods in the next chapter.

CHAPTER 5
SUBDIVISION LIMIT CALCULATIONS ON IRREGULAR SURFACE MESHES

In the context of adaptively refining meshes by directly projecting newly added boundary vertices to their exact limit position, the evaluation techniques on Loop subdivision surface meshes are used. In this chapter, we compare existing subdivision surface evaluation techniques: Jos Stam's evaluation method (Stam 1998) combined with the mirroring technique proposed by Cirak et al. (Cirak, Ortiz et al. 2000), and Persson et al.'s simplified evaluation method (Persson, Aftosmis et al. 2006). We present the advantages and disadvantages of using the related methods. In general, computing subdivision limits is particularly difficult when the corresponding boundary meshes contain extraordinary vertices or creases and hard vertices. Under this circumstance, our comparison concentrates on demonstrating the feasibility of the limit position calculations on irregular meshes.

5.1 Mirroring Technique

In Section 2.1.4, we have discussed the advantages and limitations of Jos Stam's evaluation method in details. In order to be able to evaluate any arbitrary position on any triangular surface with arbitrary topology, we integrate the mirroring technique for solving the cases with creases. The mirroring technique was cited from Schweitzer's Ph.D. thesis (Schweitzer 1996) by Cirak et al. for calculating the smoothed positions of the boundary vertices on subdivision surfaces. The mirroring technique can be explained as: temporary vertices are symmetrically mirrored along boundary edges (see Figure 5.1). For example, in Figure 5.1, two vertices x_3^0 and x_4^0 are temporarily added, boundary edge vertices $\left\{x_5^0, x_0^0, x_2^0\right\}$ and interior vertices $\left\{x_6^0, x_1^0\right\}$ are existing ones. Their relationships can be expressed by using Equation 5.1 (Cirak, Ortiz et al. 2000).

$$\begin{aligned} x_3^0 &= x_0^0 + x_2^0 - x_1^0; \\ x_4^0 &= x_0^0 + x_5^0 - x_6^0. \end{aligned} \tag{5.1}$$

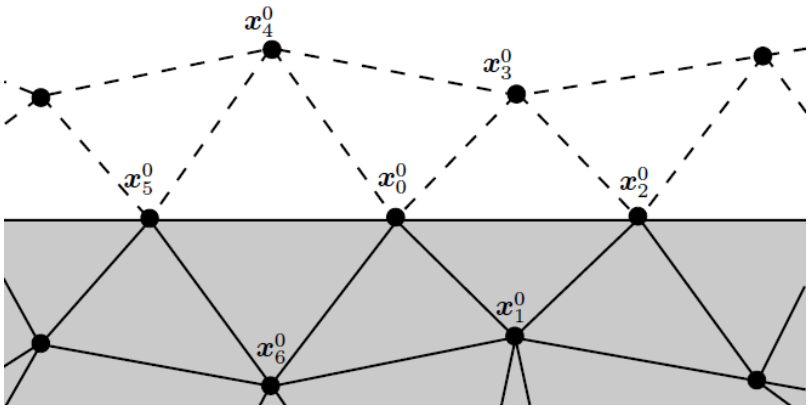

Figure 5.1: Calculating smoothed boundary edge vertices by symmetrically adding artificial vertices (Cirak, Ortiz et al. 2000).

Cirak et al. pointed out that this mirroring technique, which simplifies boundary treatment, could lead to approximation errors for irregular boundary vertices. But these errors do not prevent the convergence of the finite-element method (Cirak, Ortiz et al. 2000). In our work, we treat the defined crease edges using the mirroring technique and we deduce the position of outside vertices from existing vertices on the other sides of the crease (see Figure 5.2).

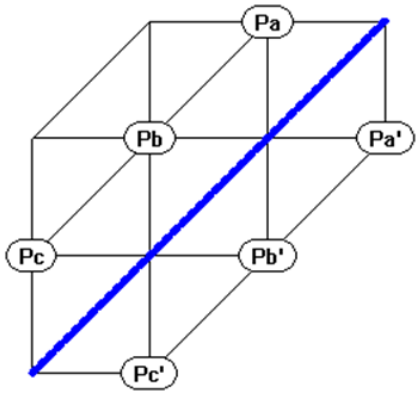

Figure 5.2: The mirroring technique applied on a regular case, where the defined edge crease is marked in large blue lines.

From Figure 5.2, we can observe that the points $\{P_a, P_b, P_c\}$ can be calculated by mirroring respectively the existing points $\{P_a, P_b, P_c\}$. This mirroring technique can be utilized for calculating vertices on boundary surfaces with the regular cases similar to the example illustrated

in Figure 5.2. In the examples above, it is easy to distinguish the two sides of the defined creases. However, edge creases are different from mesh boundaries, and sometimes their topology can be complicated (see Figure 5.3).

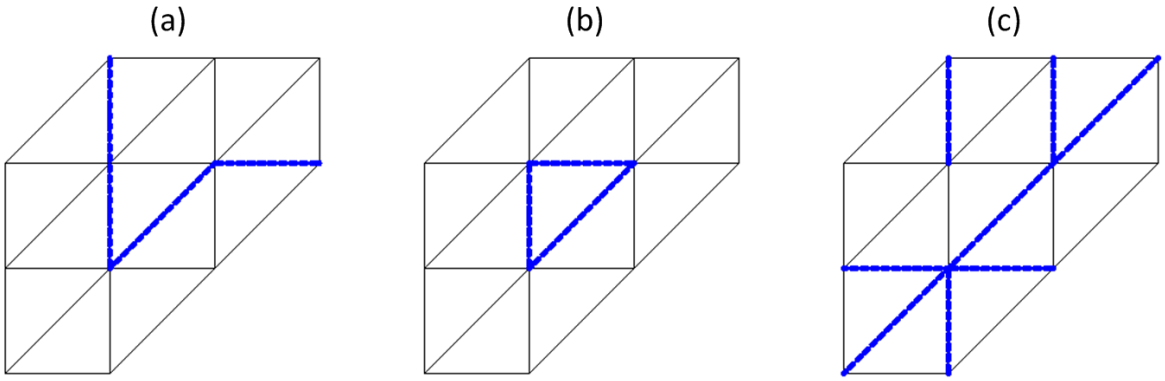

Figure 5.3: Irregular cases with complex topology, where the defined edge creases are marked in large blue lines.

From the examples in Figure 5.3, we can observe that the mirroring technique cannot be applied, since it is difficult to set the relationships between the existing points and the mirrored positions. We thought of pre-defining all possible scenarios. However, since each edge on a regular triangular patch has two states: crease or smooth and there are 24 edges on each regular triangular patch, this fact leads to 2^{24} possible scenarios. Under this circumstance, it is impossible to pre-define them.

5.2 Persson et al.'s Subdivision Surface Parameterization Method

In general, Persson et al.'s subdivision surface parameterization method (Persson, Aftosmis et al. 2006) evolves from the theory behind Jos Stam's exact evaluation method. For one irregular triangular patch, for example, the one with three extraordinary vertices (see Figure 5.4(a)), after applying the traditional subdivision iteration on it once, we obtain one refined mesh with 4 sub-triangles. As what we have presented before, one advantage of subdivision-based refinements is that they do not introduce more extraordinary vertices. Evidently, after applying this first subdivision iteration (see Figure 5.4(b)), we obtain one regular triangular patch localized in the

center refined area. This regular triangular patch (marked as green area in Figure 5.4(b)) can be directly evaluated by using the B-nets matrix expression detailed in Equation 2.20, which is presented in Section 2.1.4. Thus in Persson et al.'s method, recursive subdivisions are introduced for transforming local mesh patches into regular triangular patches. With more subdivision iterations, we can observe that the area evaluable using the B-nets matrix expression is enlarged (see Figure 5.4(c)).

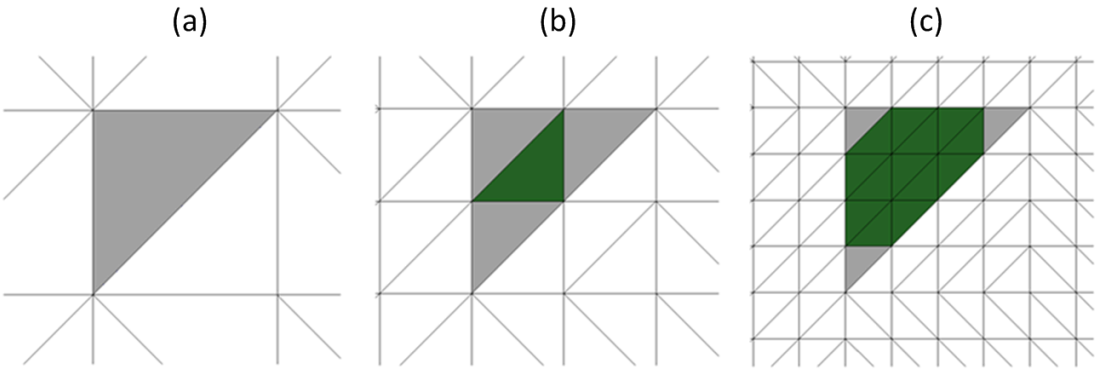

Figure 5.4: (a) one irregular triangular patch with three extraordinary vertices; (b) turning the local triangular patch (the green area) into one regular triangular patch after one subdivision-based refinement iteration; (c) enlarging the evaluable area of using the B-nets matrix expression by more subdivision iterations.

From the refined result in Figure 5.4(b), we can observe that the example above is individually localized into the three sub-irregular triangular patches shown as grey triangles, each containing only one extraordinary vertex. Thus, by applying only one subdivision iteration, any triangular patch with more than one extraordinary vertex can be evaluation by using Jos Stam's evaluation function described in Section 2.1.4. However, if the evaluation position on a triangular patch with any crease definition, the evaluation function of Jos Stam is not applicable. Whereas, using the work of Persson et al., the extraordinary vertices in the example above could be with valances not equal to 6 or with crease definitions and it would be applicable.

Persson et al. proposed a scheme where any arbitrary position on an irregular triangular patch with creases or more than one extraordinary vertex can be parameterized through simply using the B-nets matrix expression.

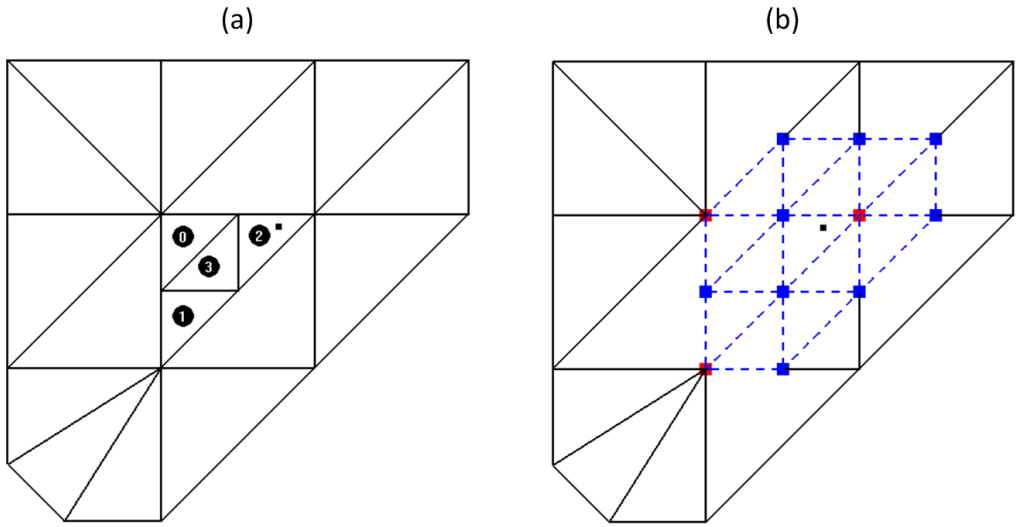

Figure 5.5: Localize the parameterized vertex in the zone, which is a regular triangular patch at the next subdivision level.

Here, we represent one arbitrary position to be parameterized as a dot in the sub-triangle 2 (see Figure 5.5(a)). To evaluate this sub-triangle, we need obtain the positions of the newly inserted edge vertices in blue and the existing vertices in red, which can be calculated by using standard Loop subdivision rules (see Figure 5.5(b)). Next, we localize the parameterized position into the restructured mesh representing the sub-triangle 2. We then apply this procedure recursively until the wanted point finally is within a regular sub-triangle. In this way we can finally apply the B-nets matrix expression.

Special treatment is to be taken to stop recursive subdivisions on the current triangular patch when the parameterized position is very close to one extraordinary vertex. In this case, the parameterized position is slightly adjusted so that it can be in the center sub-triangular patch area, which is going to be a regular sub-triangular at the next subdivision level. The error introduced from the slight adjustment is relative to the maximum recursive subdivision level

$MAX\_ITERATIONS$. From above, each triangular edge is split into two during each subdivision level. Thus, the edge length error could be $\left(\frac{1}{2}\right)^{MAX\_ITERATIONS}$. For example, if the value of $MAX\_ITERATIONS$ is set as 20 in our work, the edge length error is $\frac{\|EdgeLength\|}{1048576}$. This error is small enough to be ignored in real applications. This concrete algorithm is detailed in Table 5-1.

Table 5-1: The pseudo codes for recursively evaluating subdivision surfaces

```
EvalSurfRecursive(fasttriangle t, double u, double v, int depth, Vector3d out)
{
        if(t->isRegular())
        {
                EvalBasisDirect(t, u, v, out);
        }
        else
        {
                if(depth == MAX_RECURSION)
                {
                        //Setting the u,v in the center of the parameterized triangle t.
                        //This u, v setting facilitates in finding the regular triangle,
                        //which is usually in the center of the triangle t.
                        //Consequently it reduces the recursive iteration times.
                        u = 0.3333;
                        v = 0.3333;
                }
                fasttriangle subtriangle;
                double newu = 0.0;
                double newv = 0.0;
                SubdivideFastTriangle(t, u, v, subtriangle, newu, newv);
                EvalSurfRecursive(subtriangle, newu, newv, depth + 1, out);
        }
}
```

One drawback of this evaluation algorithm is that the CPU cost is not constant. It can vary largely depending on the inquired parametric value and the regularity of the evaluated triangle. Eventually, the cost variation is mainly decided by the number subdivision levels needed to find a regular sub-triangle.

Our improvement is to store all these evaluations in local memory (RAM), which uses only the process stack to store the sub-triangles as they are recursively generated. Due to this RAM locality fact, no cache-miss is assured and multiple evaluations can be computed in parallel. Eventually the system performance is improved.

5.3 Refinement Results related to Vertex/Edge Crease Definitions

In Section 3.1.2.2, we have presented vertex and edge crease creations on boundary surfaces of tetrahedral meshes. Regarding the discussions in Chapter 3 and Chapter 4, we restate that Loop subdivision-based tetrahedral meshes in our research are evolved. In Chapter 3, Loop subdivision tetrahedral meshes are obtained by applying the traditional uniform subdivision iterations. The solid subdivisions presented in Chapter 3 combine 1-to-8 topological splits with the Loop subdivision scheme. Differently, in Chapter 4, Loop subdivision tetrahedral meshes are obtained from using the single-level refinement method. The solid subdivisions presented in Chapter 4 combine the three different tetrahedron split patterns with the Loop subdivision limit projection. Thus, Loop subdivision-based tetrahedral meshes are evolved, from uniformly subdivided tetrahedral meshes whose smoothed boundary surfaces approach their subdivision limits, to adaptively subdivided tetrahedral meshes whose smoothed boundary surface are exactly on their subdivision limits. Consequently, the refined results integrated with crease definitions are correspondingly evolved.

In this section, we demonstrate the results of defining edge and vertex creases on boundary surfaces of tetrahedral meshes. By fixing the refinement criteria, we compare the refined results with crease definitions and without crease definitions. Two types of crease definitions: edge creases or vertex creases, are respectively marked as green edges or green vertices. We present three representative tetrahedral meshes: 1) gear (see Figure 5.6 and Figure 5.7); 2) two layer model (see Figure 5.8 and Figure 5.9) and 3) RAE2822 (airfoil extrusion) (see Figure 5.10 and Figure 5.11). The initial control meshes (displayed in red wires) of the first two examples are taken from the examples of Tetgen (Hang). The control tetrahedral mesh of RAE2822 is based on the same curve mesh of the transonic airfoil RAE2822 presented in Section 4.1.4. The start subdivision meshes of these three examples are displayed in blue purple shading mode. Regarding the visualization of adaptively refined volume interior, we display the cut profiles of the corresponding tetrahedral meshes by using the cut plane (displayed in dark red square).

In Figure 5.6, we illustrate the start tetrahedral meshes of the gear example. The illustrations of the start mesh without crease definitions from different views are displayed on the left side of Figure 5.6. Accordingly, the illustrations of the one with crease definitions are displayed on the right side of Figure 5.6. From the illustrations of Figure 5.6, we observe that the start mesh with crease definitions conserve sharp features of the gear model. The conservation of sharp features could be favorable for meeting some specific modeling requirements in numerical simulations.

In Figure 5.7, we illustrate the adaptively refined tetrahedral meshes of the gear example. We apply various refinement criteria in sequence on the same start tetrahedral mesh of the gear example: 1) the refinement criterion based on volume (its details presented in Section 4.3.4); 2) the criterion of refining Z low volume region: one volume mesh is defined in a bounding box, which is divided into six regions: X high, X low, Y high, Y low, Z high and Z low by its center. In all the following refinement examples, we specify the related region criteria as refining 30% of corresponding regions. For example, in this step, we refine 30% of Z low gear region; 3) refining Y high region; 4) refining X low region.

We illustrate the complete profiles of refined results in the top row of Figure 5.7 and respectively the cut profiles of refined results in the bottom row of Figure 5.7. From the results illustrated in Figure 5.7, the different regions of the gear model are adaptively refined. During adaptive refinements, the sharp features are kept around the defined crease regions on the refined tetrahedral mesh of the gear example.

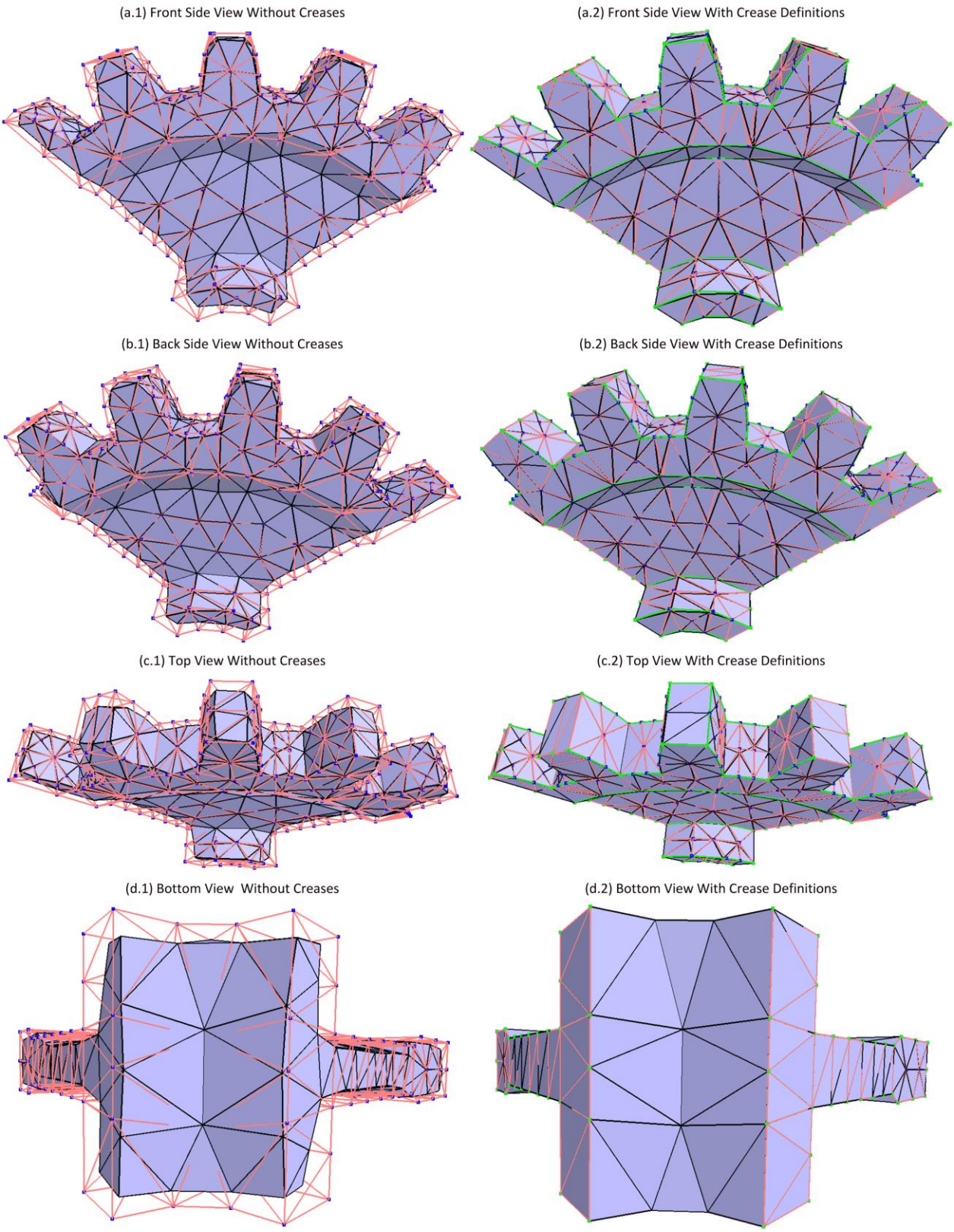

Figure 5.6: The start tetrahedral meshes of a gear from different views.

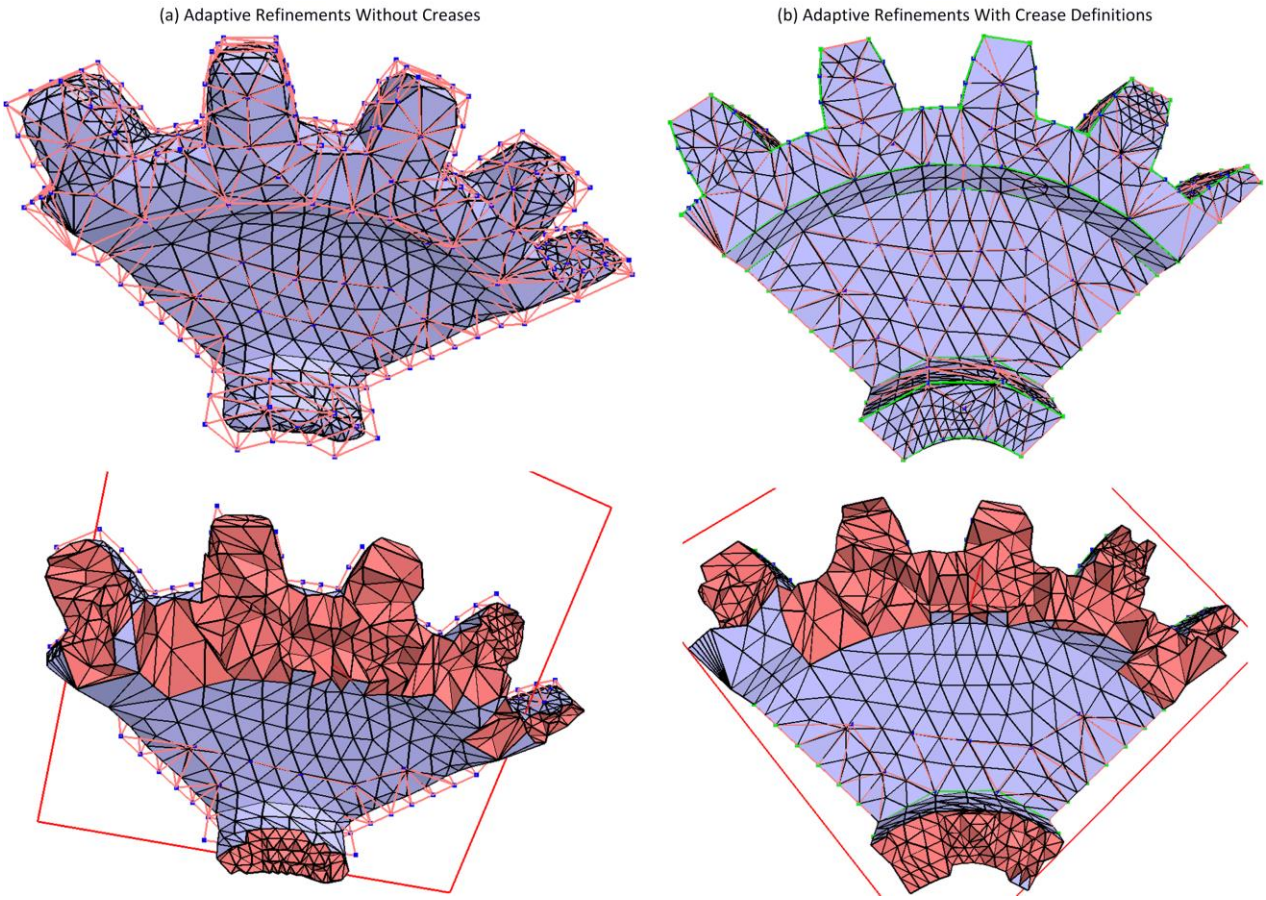

Figure 5.7: The adaptively refined tetrahedral meshes of the gear example with and without crease definitions.

In the same way, we illustrate our adaptive refinements together with crease definitions on a two layer model (see Figure 5.8 and Figure 5.9). In Figure 5.8, we display the start tetrahedral meshes of this two layer model with and without crease definitions. Then, we apply the adaptive subdivisions based on X high and Y low on the start tetrahedral meshes. In Figure 5.9, we display the refined results with and without crease definitions. From Figure 5.8 and Figure 5.9, we observe that the edge crease definitions on the two layer model facilitate preserving the original shape of the input model.

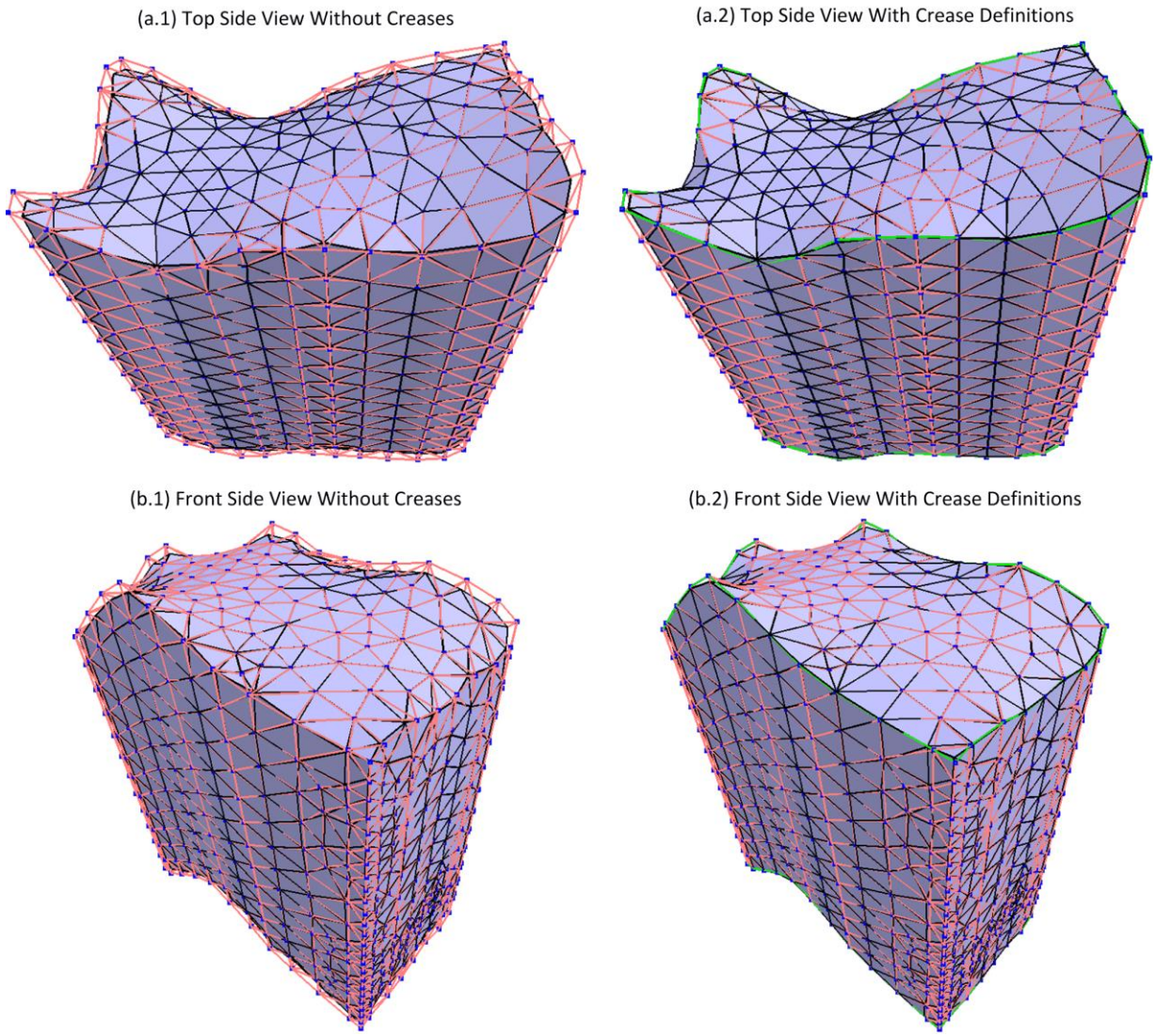

Figure 5.8: The start subdivision meshes of the two layer model with and without edge crease definitions.

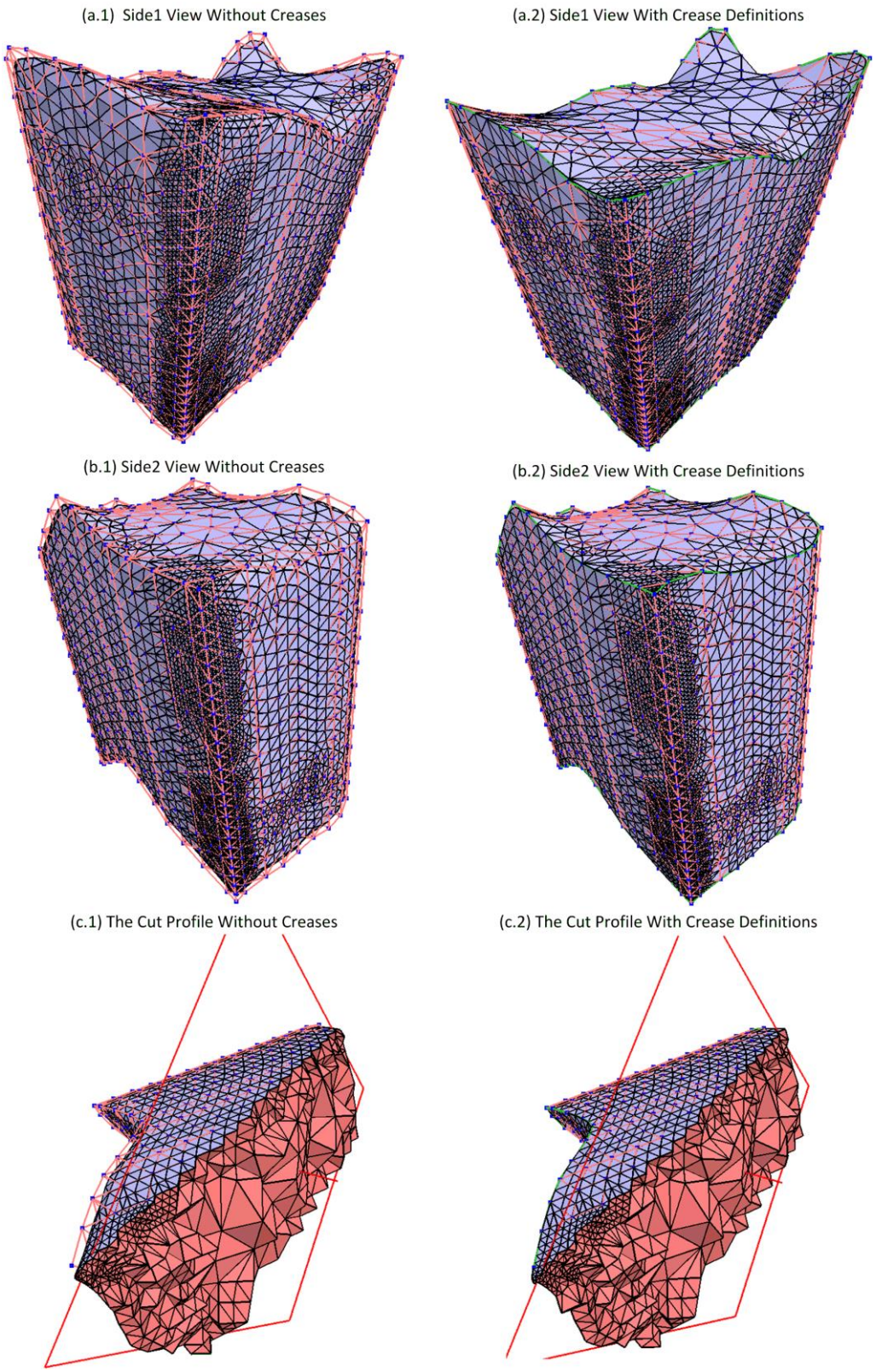

Figure 5.9: The adaptively refined subdivision meshes of the two layer model with and without crease definitions.

In Section 3.1.2.2, we have emphasized that edge crease definitions on boundary surfaces are to treat the selected edges as ‘subdivision curves’ while boundary surface are treated as subdivision surfaces during solid subdivisions. Regarding this fact, a representative example is to subdivide a tetrahedral mesh of the RAE2822 airfoil. In this example, we define the three airfoil contours as edge creases and the vertices along the airfoil trailing edge as vertex creases (see Figure 5.10(b.1) and Figure 5.10(b.2)). To better visualize crease definitions, we illustrate the start subdivision mesh of RAE2822 with and without crease definitions from different views.

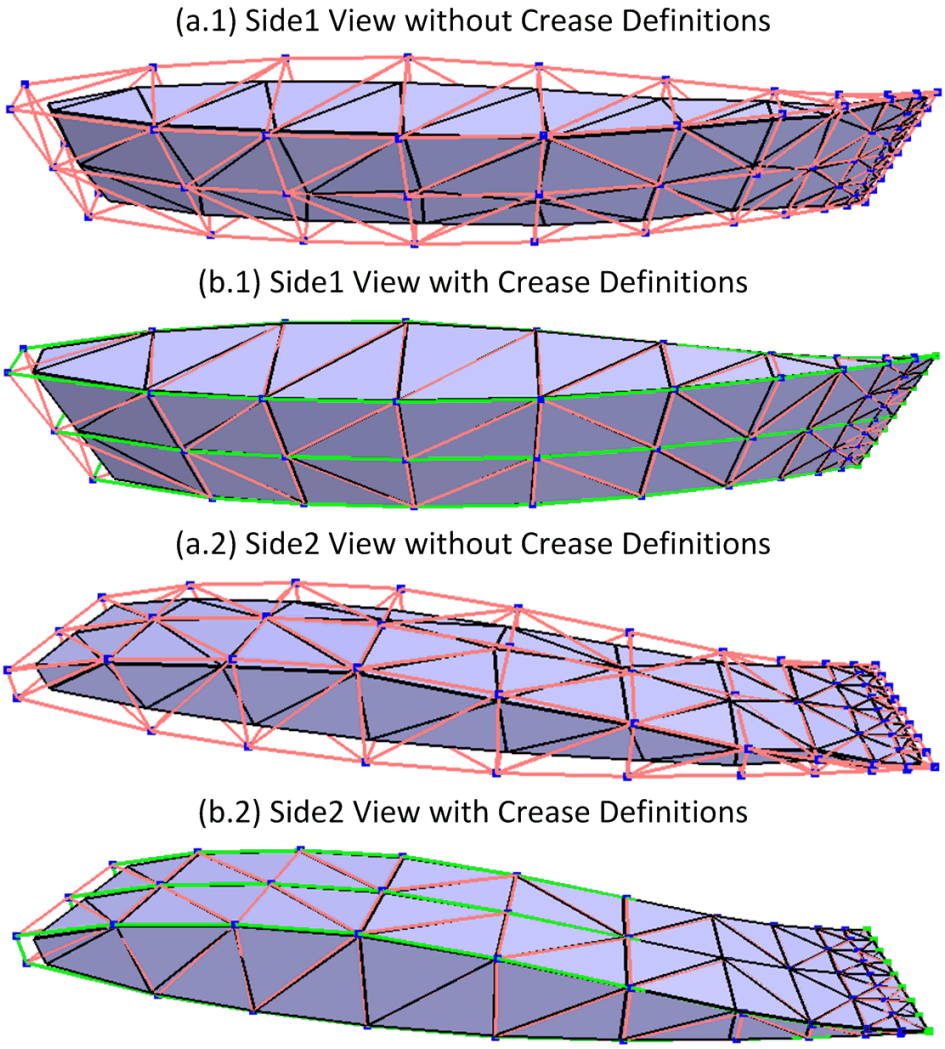

Figure 5.10: The start subdivision meshes of RAE2822 with and without crease definitions.

We apply an analytic-based criterion defined in Equation 4.8 to adaptively refine the start subdivision mesh of RAE2822 with crease definitions. Here, we set a line source as a fixed line with y coordinate equal to 0.5 and z coordinate equal to 0.13. In Figure 5.11, we respectively present the results of adaptively refining RAE2822 subdivision meshes with crease definitions by varying the value c from 1.0 to 2.0 (displayed in two sequent columns). Then according to the number of refinement iterations, in five rows we correspondingly list the refined mesh results obtained by applying the same analytic-based refinement criterion.

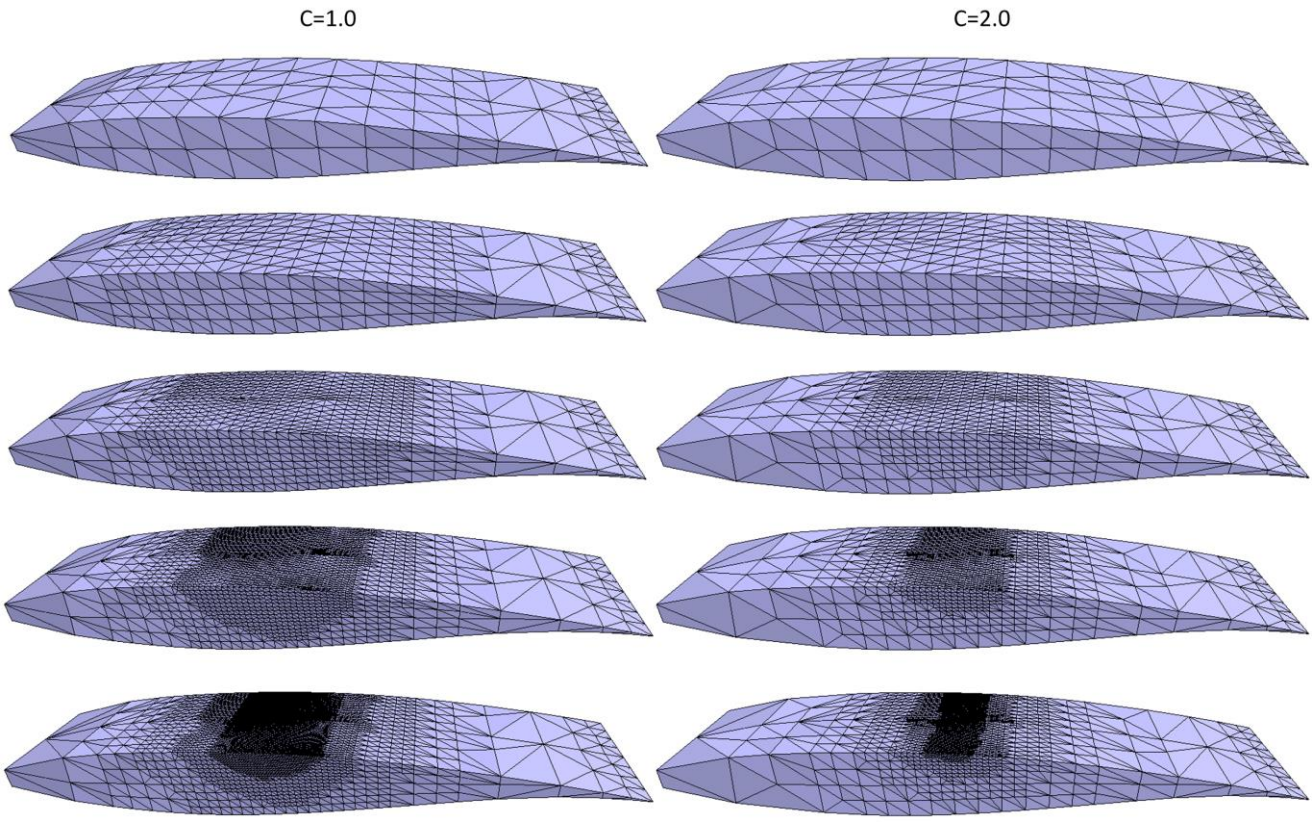

Figure 5.11: The adaptively refined subdivision meshes of RAE2822 with crease definitions by varying the constant value c.

In order to visualize the volume interior, we illustrate the cut profiles of the adaptively refined subdivision meshes of RAE2822 with crease definitions in Figure 5.12.

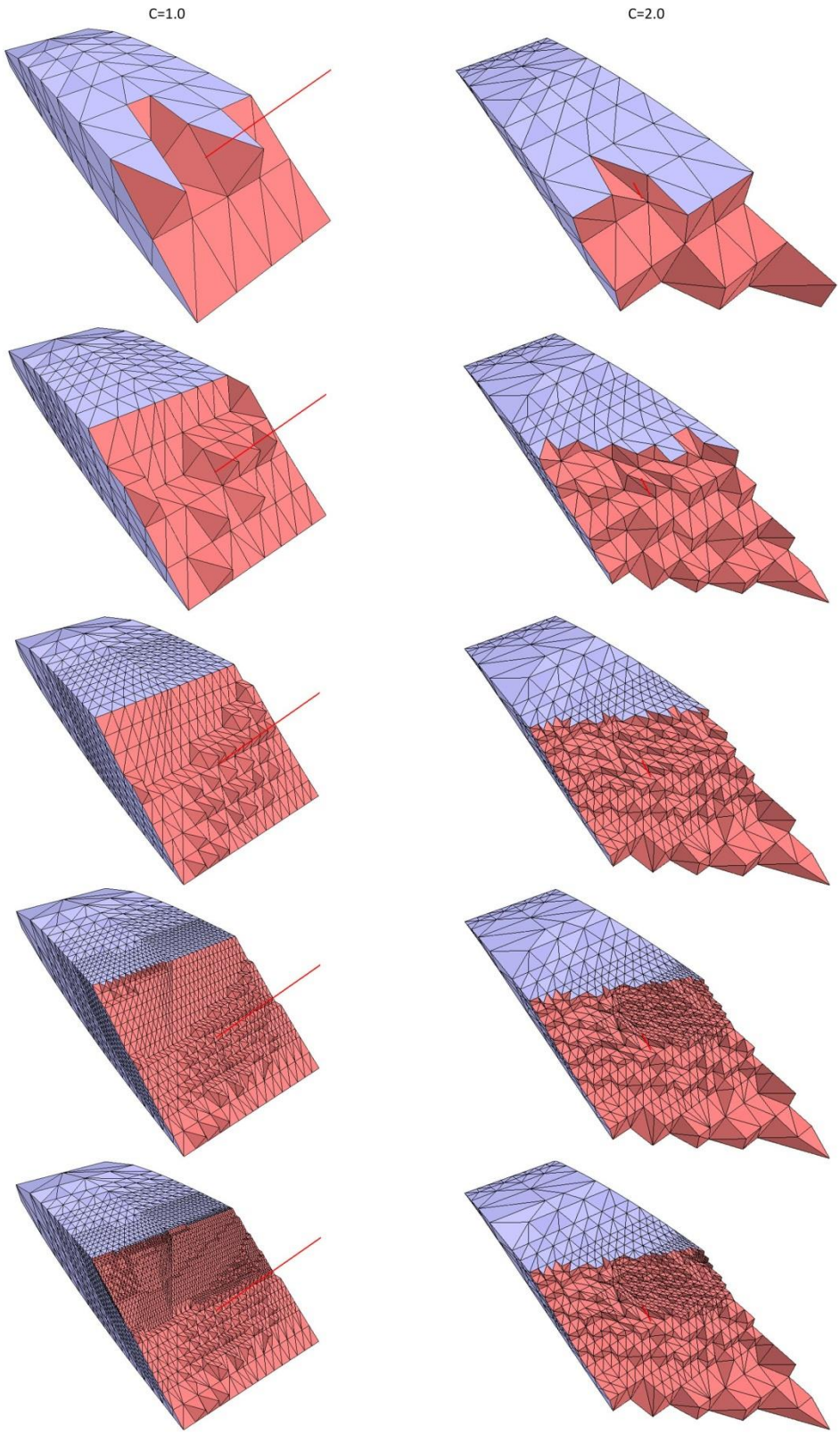

Figure 5.12: The cut profiles of the adaptively refined subdivision meshes of RAE2822 with crease definitions by varying the constant value c.

From the results of these three examples above, we observe that the two types of creases: edge crease and vertex crease are naturally integrated with adaptive subdivisions. The 3D RAE2822 example representatively demonstrates that we can obtain a multi-resolution subdivision-based representation, whose LODs can be varied by applying the same criteria on the same tetrahedral mesh. The adaptively refined results can be used for numerical simulation applications.

5.4 Evaluations of System Performance

In this section, we present the system performance evaluations regarding the efficiency issue. The evaluations are done on a PC with an Intel Core™ 2 Quad CPU at 2.4GHz. In Table 5.2, we list the time costs of successively adding new tetrahedrons during adaptive 3D refinement of the RAE2822 example with the same analytic-based refinement criterion defined in Equation 4.8. Here, we set the constant value c as 1.0.

Table 5-2: The corresponding time costs for adaptively refining the RAE2822 tetrahedral mesh with a constant value c equal to 1.0.

	Adaptive split time (seconds)	Mesh connectivity computation time (seconds)	Number of new tetrahedrons	Total number of tetrahedrons	Average tetrahedron creation time
Iteration 1	0.019118	0.013646	2809	4105	0.000012
Iteration 2	0.112397	0.169638	14919	19024	0.000019
Iteration 3	0.721609	1.922033	54897	73921	0.000048
Iteration 4	3.279184	11.157067	132085	206006	0.000109

From the results above, we observe that the time costs of generating new tetrahedrons are 12μ seconds when the number of new tetrahedrons is 2809. This time cost includes the construction of the geometry structures and data lists (tetrahedrons, faces and vertices), the subdivision limit projection of newly created edge vertices, the destruction of the removed elements (tetrahedrons and faces), the mesh connectivity computation and the calculation of the η values for the new tetrahedrons.

By studying the statistical data in the last column of Table 5-2, we remark that the average tetrahedron creation time is not constant. We use a line chart to study the relationship between the total number of tetrahedrons and the corresponding average tetrahedron creation time during each refinement iteration (see Figure 5.13). From this Figure, we observe that tetrahedron creation time is gradually increased as the corresponding mesh accuracy is increased.

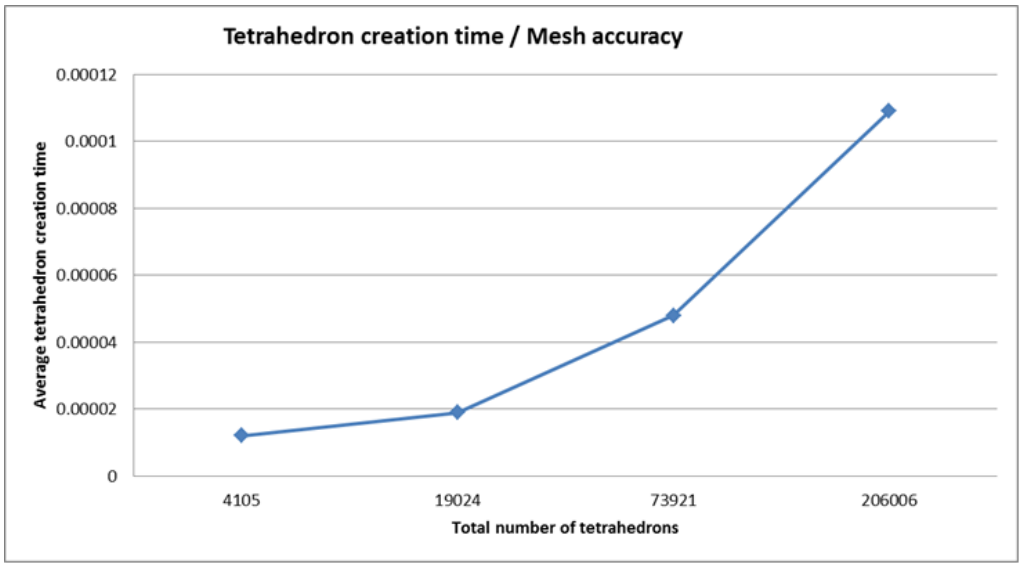

Figure 5.13: The relationship between tetrahedron creation time and mesh accuracy.

We further study the time costs related to adaptive split and mesh connectivity computation. In Figure 5.14, we observe that the time cost of adaptive splits relates to the number of the newly added tetrahedrons whereas the time cost of mesh connectivity computation relates to the total number of tetrahedrons. The time cost of mesh connectivity increases more quickly than the time cost of adaptive split calculation. These results can be explained by the fact that the mesh connectivity calculation algorithm scans the whole mesh when there is any mesh update available. This fact influences majorly system performance. This issue needs to be studied further in future.

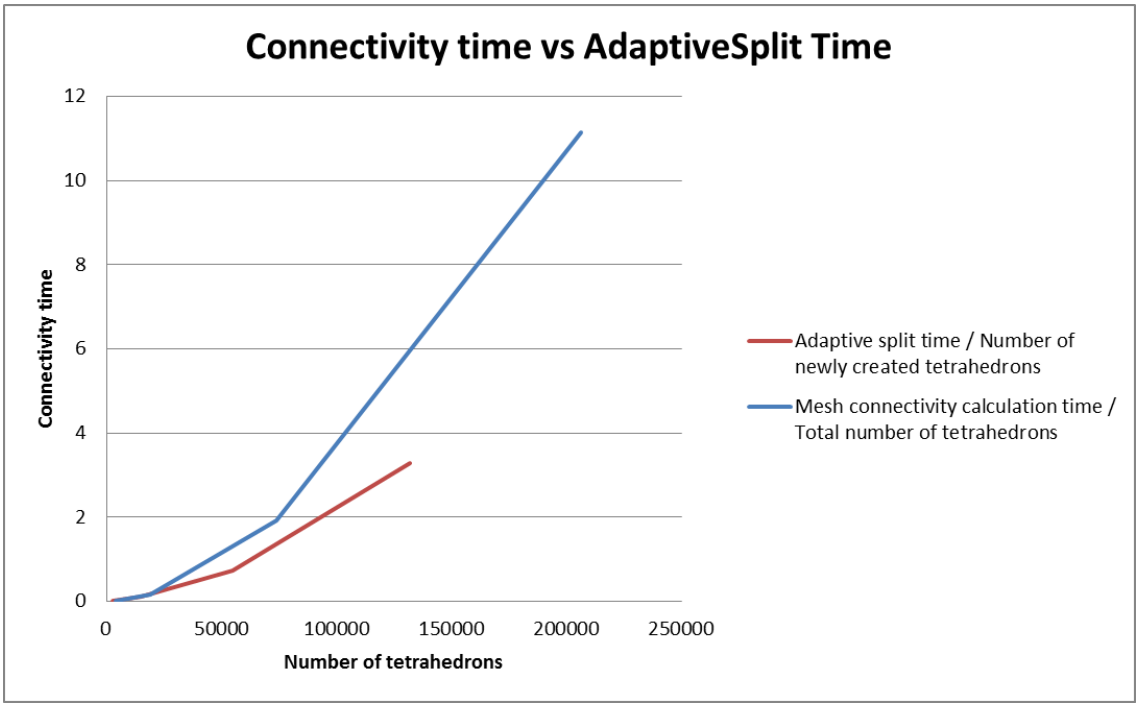

Figure 5.14: The comparison of the time costs of mesh connectivity computation and adaptive splits.

We also evaluate the time costs of calculating limit subdivision positions by using Persson's fast evaluation method on the tetrahedral meshes of the 3D RAE2822 and the gear (see Section 5.3). The time cost of calculating 11000 limit subdivision positions on the 3D RAE2822 subdivision tetrahedral mesh is 0.055 second, and the time cost of calculating 38500 limit subdivision positions on the tetrahedral mesh of the gear example is 0.191 second. Thus, the time cost of each limit position calculation is around 5.0μ second. All the results above demonstrate that our adaptive refinement work can be used for precisely and efficiently obtaining multi-resolution subdivision-based representations, even those with vertex or edge creases.

CHAPTER 6
CONCLUSION AND PERSPECTIVES

In this thesis, we outlined a novel research approach using subdivision-based representations to unify adaptive mesh refinement and geometry modification for engineering analysis purposes. The subdivision-based representations are based upon Loop subdivision techniques, more specifically, the Loop subdivision scheme and exact evaluations on Loop subdivision surfaces. This approach integrates both limit subdivision techniques for representing the underlying geometry of simulated models and adaptive subdivisions for refining meshes. Our adaptive refinement method can be used for refining curves, triangular surfaces and 3D tetrahedral meshes. The basic concept behind this refinement method can be equally used for other types of subdivision representations, for example Catmull-Clark subdivision-based representations.

This approach consists of four parts: i) developed adaptive subdivision-based refinements combining edge-based topological split patterns with geometric smoothing on boundary meshes; ii) established a single-level refinement method to support the generation of multi-resolution subdivision meshes; iii) utilized analytic geometry measurement criteria for guiding adaptive refinement; iv) evaluated the mesh quality of newly generated tetrahedral meshes and the efficiency of limit position calculation. From the obtained results, we conclude that these four aspects improved the efficiency and precision of generating multi-resolution Loop subdivision-based meshes, particularly for the ones with complex and irregular topology.

6.1 Original Contributions

The original contributions of our research approach can be summarized in three essential aspects. The first contribution of our work relates to solid subdivision and consists of two innovations: (A) A new solid subdivision scheme is developed, which is based on Loop subdivision. This Loop-based solid subdivision scheme consists of 1-to-8 topological splits and solid geometric smoothing rules. The geometric smoothing rules are specified with: 1) four standard cases regarding the existing and newly inserted vertices in the interior or on boundary surface of tetrahedral meshes; and 2) two types of crease creations on boundary surfaces: vertex and edge creases. (B) The solid subdivision is ameliorated with three tetrahedron split patterns and limit subdivision position projections on boundary surfaces. This amelioration facilitates generating

multi-resolution tetrahedral meshes, whose boundary surfaces are exactly on their subdivision limits, while their interior volumes are adaptively refined. This innovation made a significant breakthrough in the construction of subdivision-based representations.

Our single-level refinement method contributes constructive impacts to mesh adaptation in numerical simulations: Firstly, we utilize multi-resolution Loop subdivision limits as the unifying geometric representations to create, modify, discretize, analyze and visualize simulated models with arbitrary topology for numerical simulation applications. This utilization unifies mesh refinement and geometry modification, and builds a unifying foundation of the different steps of finding a numerical solution with the aid of mesh adaptation. Secondly, this method innovates upon the pure discretization-based refinement method by introducing adaptive subdivisions. The subdivision rules regarding continuous boundary mesh generation together with sharp feature preservations are naturally integrated into adaptive refinements. Thirdly, we perform adaptive subdivisions on the representative examples, for example, 3D RAE2822, a gear example, a Schoenhardt tetrahedral mesh by using analytic-based criteria. The results demonstrate that solid subdivision, adaptive refinement and analytic geometry information can be naturally integrated together for adaptively refining tetrahedral meshes. This demonstration together with the evaluations of mesh quality and system performance validate that our refinement method can be implemented in real mesh adaptation applications for meeting specific engineering analysis requirements.

In our work, we provided a technical survey of Loop subdivision surface parameterization techniques. Our contribution is to supply experimental examples for combining Jos Stam's evaluation method and Persson's work. In practice, our contribution actualizes the exact evaluation on any arbitrary position of Loop subdivision surfaces. Moreover, we extend this parameterization technique for exactly evaluating boundary surfaces of Loop subdivision-based tetrahedral meshes.

6.2 Perspectives

From above, our work involves multiple active research areas: subdivision surfaces, solid subdivision, mesh adaptation and multi-resolution modeling, where innovative approaches continuously take place. In this section, we prospect possible opportunities for future researches.

In our work, we set subdivision limits as target geometry, and then use subdivision surface parameterization to map any limit subdivision position from initial control vertices. This one-to-one mapping is achieved by supposing that initial control meshes are available. The precision of refined meshes obtained by this one-to-one mapping represents an interesting aspect, which refers us to consider the reverse mapping from subdivision limits to initial control vertices. The reverse mapping opens up a new perspective: using reverse subdivision for adaptive mesh simplification.

From the results obtained from the tetrahedron shape measure, we observe that pure subdivision as a geometric smoothing operator is not suitable for controlling mesh degeneration, but refinement criteria used for adaptive subdivision could compensate this limitation. This observation opens up another new perspective: further studying refinement criteria for obtaining mesh quality meeting specific requirements.

From the evaluation results of system performance, we remark that time for mesh connectivity computation occupy most of adaptive refinement time costs. This remark indicates that the efficiency of adaptive refinements can be improved further by optimizing the mesh connectivity computation algorithm. This indication eventually opens up another new perspective.

BIBLIOGRAPHIE

Alliez, P., G. Ucelli, C. Gotsman and M. Attene (2005). Recent Advances in Remeshing of Surfaces, AIM@SHAPE Network of Excellence.

Aspert, N., D. Santa-Cruz and T. Ebrahimi (2002). MESH: measuring errors between surfaces using the Hausdorff distance. Proceedings of IEEE International Conference on Multimedia and Expo (ICME), Lausanne, Switzerland, IEEE.

Bajaj, C., S. Schaefer, J. Warren and X. Guoliang (2002). "A subdivision scheme for hexahedral meshes." Visual Computer **18**(5-6): 343-356.

Barthe, L. (2005). Box-Splines et Surfaces de Subdivision. AFIG. Strasbourg, France.

Biermann, H., I. Martin, F. Bernardini and D. Zorin (2002). Cut-and-paste editing of multiresolution surfaces. SIGGRAPH '02: 29th International Conference on Computer Graphics and Interactive Techniques, San Antonio, TX, ACM.

Biermann, H., I. M. Martin, D. Zorin and F. Bernardini (2002). "Sharp features on multiresolution subdivision surfaces." Graphical Models **64**(2): 61-77.

Burkhart, D., B. Hamann and G. Umlauf (2010). "Adaptive and Feature-Preserving Subdivision for High-Quality Tetrahedral Meshes." Computer Graphics Forum **29**(1): 117-127.

Burkhart, D., B. Hamann and G. Umlauf (2010). "Iso-geometric Finite Element Analysis Based on Catmull-Clark Subdivision Solids." Computer Graphics Forum **29**(5): 1575-1584.

Burkhart, D., B. Hamann and G. Umlauf (2011). Finite element analysis for linear elastic solids based on subdivision schemes. International Research and Training Group 1131 Workshop on Visualization of Large and Unstructured Data Sets - Applications in Geospatial Planning, Modeling and Engineering, VLUDS 2010, March 19, 2010 - March 21, 2010, Bodega Bay, CA, United states, Schloss Dagstuhl - Leibniz-Zentrum fur Informatik GmbH.

Catmull, E. and J. Clark (1978). "Recursively generated B-spline surfaces on arbitrary topological meshes." Computer Aided Design **10**(6): 350-355.

Chaikin, G. (1974). "An Algorithm for High Speed Curve Generation." Computer Graphics and Image Processing(3): 346-349.

Chang, Y.-S., K. T. McDonnell and H. Qin (2002). "A new solid subdivision scheme based on box splines." Proceedings of the Symposium on Solid Modeling and Applications: 226-233.

Chang, Y.-S., K. T. McDonnell and H. Qin (2003). An interpolatory subdivision for volumetric models over simplicial complexes. Proceedings. Shape Modeling International 2003, Seoul, South Korea, IEEE Comput. Soc.

Cignoni, P., C. Montani and R. Scopigno (1998). "A comparison of mesh simplification algorithms." Computers & Graphics **22**(1): 37-54.

Cignoni, P., C. Rocchini and R. Scopigno (1998). "Metro: measuring error on simplified surfaces." Computer Graphics Forum **17**(2): 167-174.

Cirak, F., M. Ortiz and P. Schroder (2000). "Subdivision surfaces: a new paradigm for thin-shell finite-element analysis." International Journal for Numerical Methods in Engineering **47**(Copyright 2000, IEE): 2039-2072.

Cirak, F., M. J. Scott, E. K. Antonsson, M. Ortiz and P. Schroder (2002). "Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision." Computer Aided Design **34**(2): 137-148.

Clark, J. H. (1976). "Hierarchical geometric models for visible surface algorithms." Communications of the ACM **19**(10): 547-554.

Corney, J., C. Hayes, V. Sundararajan and P. Wright (2005). "The CAD/CAM Interface: A 25-Year Retrospective." Journal of Computing and Information Science in Engineering **5**(3): 188-197.

De Boor, C. (2001). A practical guide to splines. New York, Springer.

De Floriani, L. and P. Magillo (2002). Multiresolution mesh representation: Models and data structures. Tutorials on multiresolution in geometric modelling : summer school lectures notes. A. Iske, E. Quak and M. S. Floater. Berlin ; New York, Springer-Verlag**:** 363-418.

Dodgson, N. A., U. H. Augsdorfer, T. J. Cashman and M. A. Sabin (2009). Deriving box-spline subdivision schemes. Mathematics of Surfaces XIII. 13th IMA International Conference, 7-9 Sept. 2009, Berlin, Germany, Springer-Verlag.

Dompierre, J., M. G. Vallet, P. Labbe and F. Guibault (2005). "An analysis of simplex shape measures for anisotropic meshes." Computer Methods in Applied Mechanics and Engineering **194**(48-49): 4895-4914.

Doo, D. and M. Sabin (1978). "Behaviour of recursive division surfaces near extraordinary points." Computer Aided Design **10**(6): 356-360.

Fidkowski, K. J. and D. L. Darmofal (2011). "Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics." AIAA Journal **49**(4): 673-694.

Garland, M. (1999). Multiresolution Modeling: Survey & future opportunities, State of the Art Report, Eurographics ’99.

Garland, M. (1999). Quadric-based polygonal surface simplification Ph.D., Carnegie Mellon University.

Garland, M. and Y. Zhou (2005). "Quadric-based simplification in any dimension." Acm Transactions on Graphics **24**(2): 209-239.

Gonsor, D. and M. Neamtu (2001). Subdivision surfaces -- can they be useful for geometric modeling applications., Boeing Company.

Griessmair, J. and W. Purgathofer (1989). Deformation of solids with trivariate B-splines. EUROGRAPHICS '89. Proceedings of the European Computer Graphics Conference, Hamburg, West Germany, North-Holland.

Gursoy, H. N. (1996). "Tetrahedral finite element mesh generation from NURBS solid models." Engineering with Computers **12**(3-4): 211-223.

Guskov, I. (2006). "Manifold-based approach to semi-regular remeshing." Graphical Models **69**(1): 1-18.

Guskov, I., A. Khodakovsky, P. Schröder and W. Sweldens (2002). Hybrid meshes: multiresolution using regular and irregular refinement. Proceedings of the eighteenth annual symposium on Computational geometry, Barcelona, Spain, ACM.

Hang, S. "TetGen: A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator ", from http://wias-berlin.de/software/tetgen/.

Hassan, M. F. and N. A. Dodgson (2005). Reverse Subdivision. Advances in Multiresolution for Geometric Modelling. N. A. Dodgson, M. S. Floater and M. A. Sabin, Springer**:** 283-271.

Hoppe, H., T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer and W. Stuetzle (1994). Piecewise smooth surface reconstruction. Proceedings of the Annual Conference on Computer Graphics, Jul 24 - 29 1994, Orlando, FL, United states, ACM.

ISO (1994). Industrial automation systems and integration (International Organization for Standardization). Product data representation and exchange. Geneva, Switzerland. **10303**.

Ito, Y., A. M. Shih and B. K. Soni (2009). "Octree-based reasonable-quality hexahedral mesh generation using a new set of refinement templates." International Journal for Numerical Methods in Engineering **77**(13): 1809-1833.

Joy, K. I. (1991). Utilizing parametric hyperpatch methods for modeling and display of free-form solids. Proceedings. Symposium on Solid Modeling Foundations and CAD/CAM Applications, Austin, TX, ACM.

Joy, K. I. and M. A. Duchaineau (1999). Boundary determination for trivariate solids. Proceedings. Seventh Pacific Conference on Computer Graphics and Applications, Seoul, South Korea, IEEE Comput. Soc.

Junye, W., Z. Xiaoxian, A. G. Bengough and J. W. Crawford (2008). "Performance evaluation of the cell-based algorithms for domain decomposition in flow simulation." International Journal of Numerical Methods for Heat & Fluid Flow **18**(5): 656-672.

Kerlow, I. V. (2004). The art of 3D computer animation and effects. Hoboken, N.J., John Wiley & Sons.

Kobbelt, L. (2000). Sqrt(3)-subdivision. Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co.**:** 103-112.

Kobbelt, L., S. Campagna, J. Vorsatz and H.-P. Seidel (1998). Interactive multi-resolution modeling on arbitrary meshes. Proceedings of the 25th annual conference on Computer graphics and interactive techniques, ACM**:** 105-114.

Labbé, P., F. Guibault, J. Dompierre, M. G. Vallet and J.-Y. Trépanier (2001). A Generic Mesher for STEP Compliant Geometries. Proceedings of the 48th Annual CASI Conference, Toronto, Ontario, Canadian Aeronautics and Space Institute.

Lanquetin, S. and M. Neveu (2006). Reverse Catmull-Clark subdivision. 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2006, WSCG'2006 - In Co-operation with EUROGRAPHICS, January 31, 2006 - February 2, 2006, Plzen, Czech republic, Vaclav Skala Union Agency.

Lasser, D. (1985). Bernstein-Bezier representation of volumes. Surfaces in CAGD '84, Oberwolfach, West Germany.

Loop, C. (1987). Smooth Subdivision Surfaces Based on Triangles. M.S., University of Utah.

Lounsbery, M., T. D. DeRose and J. Warren (1997). "Multiresolution analysis for surfaces of arbitrary topological type." ACM Transactions on Graphics **16**(1): 34-73.

Luebke, D., M. Reddy, J. D. Cohen, A. Varshney, B. Watson and R. Hubener (2002). Level of detail for 3D graphics. Amsterdam, Morgan Kaufmann.

Ma, W. (2005). "Subdivision surfaces for CAD - an overview." Computer Aided Design **37**(7): 693-709.

MacCracken, R. and K. I. Joy (1996). "Free-form deformations with lattices of arbitrary topology." Proceedings of the ACM SIGGRAPH Conference on Computer Graphics: 181-188.

Mallat, S. G. (1989). "A theory for multiresolution signal decomposition: the wavelet representation." IEEE Transactions on Pattern Analysis and Machine Intelligence **11**(7): 674-693.

Marinov, M. and L. Kobbelt (2005). "Optimization methods for scattered data approximation with subdivision surfaces." Graphical Models **67**(5): 452-473.

Martin, T., E. Cohen and M. Kirby (2008). Volumetric parameterization and trivariate b-spline fitting using harmonic functions. Proceedings of the 2008 ACM symposium on Solid and physical modeling. Stony Brook, New York, ACM**:** 269-280.

Meyer, Y. (1993). Wavelets and Operators, Cambridge University Press.

Mezentsev, A. (2007). An efficient geometrical model for meshing applications in heterogeneous environments. Proceedings of the 16th International Meshing Roundtable.

Mezentsev, A. and T. Woehler (1999). Methods And Algorithms Of Automated CAD Repair For Incremental Surface Meshing. Proceedings of the 8th International Meshing Roundtable, South Lake Tahoe, USA.

Ming-Jun, L. (1992). "Fortran subroutines for B-nets of box splines on three- and four-directional meshes." Numerical Algorithms **2**(1): 33-38.

Muller, H. and R. Jaeschke (1998). Adaptive subdivision curves and surfaces. Proceedings Computer Graphics International, 22-26 June 1998, Los Alamitos, CA, USA, IEEE Comput. Soc.

Owen, S. J. (2005). An Introduction to Mesh Generation Algorithms. 14th international meshing roundtable. San Diego, California, USA.

Owen, S. J., D. R. White and T. J. Tautges (2002). Facet-based Surfaces for 3d Mesh Generation. Proceeding 11th International Meshing Roundtable.

Pajarola, R. and E. Gobbetti (2007). "Survey of semi-regular multiresolution models for interactive terrain rendering." Visual Computer **23**(8): 583-605.

Persson, P.-O., M. J. Aftosmis and R. Haimes (2006). On The Use of Loop Subdivision Surfaces for Surrogate Geometry. the 15th International Meshing Roundtable.

Piegl, L. and W. Tiller (1997). The NURBS book (2nd ed.), Springer-Verlag New York, Inc.

Pólya, G. (1964). Mathematics and Plausible Reasoning: Patterns of plausible inference.

Qian, J. and Y. Zhang (2010). Sharp Feature Preservation in Octree-Based Hexahedral Mesh Generation for CAD Assembly Models. Proceedings of the 19th International Meshing Roundtable. S. Shontz, Springer Berlin Heidelberg**:** 243-262.

Qian, J. and Y. Zhang (2011). "Automatic unstructured all-hexahedral mesh generation from B-Reps for non-manifold CAD assemblies." 1-15.

Rossignac, J. R. and A. A. G. Requicha (1999). Solid Modeling. J. Webster. Encyclopedia of Electrical and Electronics Engineering.

Roy, M., S. Foufou, A. Koschan, F. Truchetet and M. Abidi (2006). Multiresolution analysis for meshes with appearance attributes. 2005 International Conference on Image Processing, Genova, Italy, IEEE.

Roy, M., F. Nicolier, S. Foufou, F. Truchetet, A. Koschan and M. A. Abidi (2002). Assessment of mesh simplification algorithm quality. Three-Dimensional Image Capture and Applications V, San Jose, CA, SPIE-Int. Soc. Opt. Eng.

Sadeghi, J. and F. F. Samavati (2011). "Smooth reverse Loop and Catmull-Clark subdivision." Graphical Models **73**(5): 202-217.

Schaefer, S. Linear subidivision followed by averaging on a square subdivision.org.

Schroder, P. (1999). Opportunities for subdivision-based multiresolution modeling. Computer Graphics and Applications, 1999. Proceedings. Seventh Pacific Conference on.

Schroder, P. (2004). Subdivision: Smooth Bases on Meshes (and Other Neat Tricks)**:** Oberwolfach Seminar on Discrete Differential Geometry.

Schweitzer, J. E. (1996). Analysis and application of subdivision surfaces. 9704545 Ph.D., University of Washington.

Sederberg, T. W. and S. R. Parry (1986). Free-form deformation of solid geometric models. SIGGRAPH '86 Conference Proceedings, Dallas, TX.

Seungyong, L. (1999). Interactive multiresolution editing of arbitrary meshes. European Association for Computer Graphics 20th Annual Conference. EUROGRAPHICS'99, 7-11 Sept. 1999, UK, Blackwell Publishers for Eurographics Assoc.

Shaffer, E. and M. Garland (2005). "A Multiresolution Representation for Massive Meshes." IEEE Transactions on Visualization and Computer Graphics **11**(2): 139-148.

Shewchuk, J. R. (1998). Tetrahedral mesh generation by Delaunay refinement. Proceedings of SCG 98: 14th ACM Symposium on Computational Geometry, 7-10 June 1998, New York, NY, USA, ACM.

Silva, F. G. M. and A. J. P. Gomes (2003). Adjacency and incidence framework: a data structure for efficient and fast management of multiresolution meshes. Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia (SESSION: Meshes), Melbourne, Australia, ACM.

Stam, J. (1998). Evaluation of Loop Subdivision Surfaces. Computer Graphics Proceedings ACM SIGGRAPH 1998.

Stam, J. (1998). Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. Proceedings of the 25th annual conference on Computer graphics and interactive techniques, ACM**:** 395-404.

Sunil, V. B., R. Agarwal and S. S. Pande (2010). "An approach to recognize interacting features from B-Rep CAD models of prismatic machined parts using a hybrid (graph and rule based) technique." Comput. Ind. **61**(7): 686-701.

Surazhsky, V. and C. Gotsman (2005). A Qualitative Comparison of Some Mesh Simplification Software Packages.

UIUC. "The UIUC Airfoil Data Site." from http://www.ae.illinois.edu/m-selig/ads/coord/rae2822.dat.

Volkov, V. and L. Ling (2003). Real-time refinement and simplification of adaptive triangular meshes. IEEE Visualization 2003, 19-24 Oct. 2003, Piscataway, NJ, USA, IEEE.

Wang, M. (2006). A multiresolution subdivision-based framework for interactive surface mesh editing. Master, École polytechnique de Montréal.

Warren, J. and H. Weimer (2002). Subdivision Methods for Geometric Design, Morgan Kaufmann.

wikipedia.org. "Curvature." from http://en.wikipedia.org/wiki/Curvature.

Wikipedia.org. "Octree." from http://en.wikipedia.org/wiki/Octree.

Wikipedia.org. "Regular Tetrahedron." from http://en.wikipedia.org/wiki/Regular_tetrahedron.

Yang, J. and S. Han (2006). "Repairing CAD model errors based on the design history." Computer-Aided Design **38**(6): 627-640.

Zhang, Y., Y. Bazilevs, S. Goswami, C. L. Bajaj and T. J. R. Hughes (2007). "Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow." Computer Methods in Applied Mechanics and Engineering **196**(29-30): 2943-2959.

Zorin, D. (2006). Modeling with multiresolution subdivision surfaces. ACM SIGGRAPH 2006 Courses. Boston, Massachusetts, ACM**:** 30-50.

Zorin, D., M. Holst and P. Schroder (1997). Subdivision-based surface representations. Proceedings of TeamCAD: 1st GVU Workshop on Collaborative Design, Atlanta, GA, Georgia Inst. Technol.

Zorin, D., P. Schroder, T. Derose, L. Kobbelt, A. Levin and W. Sweldens (2000). Subdivision for animation and modeling. SIGGRAPH 2000.

Zorin, D., P. Schroder and W. Sweldens (1997). Interactive multiresolution mesh editing. Proceedings of 24th International Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, ACM.